DOCUMENT RESUME

ED 022 677 SE 004 693

By-Feng, Chuan C.

COMPUTER RELATED MATHEMATICS AND SCIENCE CURRICULUM MATERIALS - A NATIONAL SCIENCE FOUNDATION COOPERATIVE COLLEGE-SCHOOL SCIENCE PROGRAM IN COMPUTING SCIENCE EDUCATION.

Boulder Valley School District, Colo.; Jefferson County School District, Colo.

Spons Agency-National Science Foundation, Washington, D.C.

Report No-GW-1755

Pub Date 67

Note-236p.

EDRS Price MF-$1.00 HC-$9.52

Descriptors-ALGEBRA, *COMPUTER ORIENTED PROGRAMS, COMPUTER PROGRAMS, *CURRICULUM DEVELOPMENT, INSTRUCTION, MATHEMATICS, NUMBER CONCEPTS, PHYSICS, SCIENCES, *SECONDARY SCHOOL MATHEMATICS, *SECONDARY SCHOOL SCIENCE, TEACHER EDUCATION

Identifiers-Cooperative College-School Science Program, National Science Foundation, University of Colorado

Reported is the Cooperative College-School Science Program in Computing Science Education which was conducted by the University of Colorado Department of Civil Engineering in the summer of 1967. The program consisted of two five-week terms. The course work was composed of two formal lecture courses in Computer Related Mathematics and Computer Programing. Laboratory work encompassed demonstrations, discussion periods, and problem solving by the 40 mathematics and science secondary school teachers using BASIC language on remote consoles. This document is a compilation of some special projects undertaken by teachers (individually or in pairs), assisted in some cases by some 24 eleventh grade students who participated in the program. These special projects consisted of writing the details of typical high school lectures that incorporate computer related concepts. The main objective of the special projects was to begin to direct the high school teacher toward computer-oriented educational materials. (RP)

U.S. DEPARTMENT OF HEALTH, EDUCATION & WELFARE
OFFICE OF EDUCATION

THIS DOCUMENT HAS BEEN REPRODUCED EXACTLY AS RECEIVED FROM THE PERSON OR ORGANIZATION ORIGINATING IT. POINTS OF VIEW OR OPINIONS STATED DO NOT NECESSARILY REPRESENT OFFICIAL OFFICE OF EDUCATION POSITION OR POLICY.

COMPUTER RELATED MATHEMATICS AND SCIENCE CURRICULUM MATERIALS

by
Teacher Participants
Boulder Valley and Jefferson County
School Districts

R-1

1876

ED022677

SE 004 693

A publication of a Cooperative
College-School Science Program
in Computing Science Education
funded by the National Science
Foundation under Grant GW-1755
and held at the University of
Colorado, Summer, 1967.

A National Science Foundation
Cooperative College - School Science
Program In
Computing Science Education
University of Colorado
Summer 1967

COMPUTER RELATED MATHEMATICS AND SCIENCE CURRICULUM MATERIALS

by

Teacher Participants
Boulder Valley and Jefferson County
School Districts

FACULTY AND STAFF

Chuan C. Feng
Professor
Univ. of Colorado
Program Director

Larry J. Feeser
Associate Professor
Univ. of Colorado
Program Co-Director

Frederic O. Woodsome
Associate Professor
Univ. of Colorado
Program Co-Director

R. Thomas Clement
Program Instructor

Edwin Olmsted
Program Laboratory Associate

Ronald Baker
Program Laboratory Associate

Warren Burton
Program Laboratory Assistant

SPECIAL LECTURERS

Robert E. Albrecht
Educational Consultant

Harold H. Greenberg
Systems Engineer
General Electric Company

William H. Huggins
Westinghouse Professor of
Electrical Engineering
The Johns Hopkins University

George W. Morgenthaler
Director of Research and Development
Denver Division
Martin-Marietta Corporation

George Morosow
Chief, Dynamics and Loads
Denver Division
Martin-Marietta Corporation

ADVISORY GROUP

Robert S. Ayre
Professor and Chairman
Civil Engineering
University of Colorado

Paul E. Bartlett
Associate Dean, Engineering
Univ. of Colorado

Leslie M. Chase
Deputy Superintendent
Jefferson County School District

Richard M. Fawley
Acting Superintendent
Boulder Valley School District

EDITORIAL ADVISORS

Ralph E. Bachus
Science Supervisor
Boulder Valley School District

John W. Bradford
Mathematics Coordinator
Jefferson County School District

Glenn A. Gilbert
Mathematics Supervisor
Boulder Valley School District

Siegfried Mandel
Professor, English in Engineering
University of Colorado

TEACHER PARTICIPANTS

Ralph Bachus
Boulder Valley School District

Robert L. Bassett
Arvada High School

Mildred J. Beavers
Boulder High School

John W. Bradford
Jefferson County School District

Robert L. Brown
Boulder High School

Dewayne N. Burgdorff
Longmont High School

Milton R. Collum
Drake Junior High School

Dan J. Colvin
Jefferson High School

Wayne B. Daniels
Casey Junior High School

William E. Einert
Boulder High School

Sandra L. Epstein
Bear Creek High School

Carol E. Friend
Wheatridge High School

Cheryl L. Gaskell
Jefferson High School

Glenn A. Gilbert
Boulder Valley School District

Margaret A. Grill
Southern Hills Junior High School

John G. Groninger, Jr.
Alameda High School

Neil O. Haflich
Baseline Junior High School

Marion E. Hakewessell
Alameda High School

Harry A. Hurley
Burbank Junior High School

Hepburn Ingham
Lafayette High School

Royce D. Jackson
Evergreen High School

Dan H. Jarrell
Arvada West High School

Nick R. Kalianov
Alameda High School

Dean C. Larsen
Bear Creek High School

Donald C. Lohrentz
Lakewood High School

Kenneth W. Mahan
Alameda Junior High School

Donald H. Marcotte, Jr.
North Arvada Junior High School

Kenneth A. Marine
Everritt Junior High School

Ellis R. Mercer
Baseline Junior High School

Marvin L. Miller
Lakewood Junior High School

Jay E. Niebur
Nederland High School

Robert A. Norton
Baseline Junior High School

Raymond A. Platt
Wheatridge Junior High School

Pearl I. Price
Lakewood High School

Helen E. Schweizer
Burbank Junior High School

Terry E. Shoemaker
Alameda High School

Billy L. Tipton
Arvada High School

Roe E. Willis
Burbank Junior High School

Lavere C. Wilson
Fairview High School

Karen L. Winquist
Fairview High School

STUDENT PARTICIPANTS

Cheryl J. Anderson
Alameda High School

Jim O. Beehler
Boulder High School

Lewis T. Bruins
Boulder High School

Gary L. Carpenter
Wheatridge High School

Richard L. Cody
Arvada High School

Joel S. Forrest
Alameda High School

John A. Haskovec
Boulder High School

Vicky A. Isakson
Boulder High School

Jolene S. Landers
Longmont High School

David W. Larson
Lakewood High School

Martha Lodge
Boulder High School

Theo Scott Mandel
Boulder High School

Chris C. Maurer
Boulder High School

Robert J. McCarty
Longmont High School

Hanferd J. Moen, Jr.
Arvada West High School

Allan P. Mueller
Boulder High School

Bruce G. Oldaker
Arvada West High School

Linda O'Neil
Boulder High School

David F. Oslund
Bear Creek High School

Steven A. Schmidt
Jefferson High School

Gary A. Tubb
Lakewood High School

Gwendolyn R. Valdeck
Bear Creek High School

Everett W. Van Westenberg
Arvada High School

Jeffrey S. Wagener
Fairview High School

FOREWORD

During the summer of 1967, A Cooperative College-School Science Program in Computing Science Education was conducted by the University of Colorado Department of Civil Engineering with the support of the National Science Foundation (Grant GW 1755). The program consisted of two 5-week terms. The first term was held in cooperation with the Boulder Valley School District; the second term, in cooperation with the Jefferson County School District. Twenty mathematics and/or science secondary school teachers and twelve eleventh grade students participated during each term.

The course content for the two terms was identical and consisted of course work, laboratory work, and study periods each day. The coursework was composed of two formal lecture courses -- Computer Related Mathematics and Computer Programming; while the laboratory work encompassed demonstrations, discussion periods, and problem solving by the participants using BASIC language on remote consoles connected to the General Electric time-sharing computer system in Phoenix, Arizona.

During each term, many of the teachers (individually or in pairs), assisted in some cases by students, undertook special projects. These special projects consisted of writing up the details of typical high school lectures, incorporating computer related concepts. In some cases, the topics chosen were those ordinarily presented, with modification to reflect a computer oriented approach. In other cases, the topics were new computer related ones that could replace certain elements in an existing course. This document is a compilation of these special projects.

The main objective of the special projects was to begin to motivate the high school teacher's development toward computer oriented educational materials. Hopefully, by printing all projects, individual teachers will be able to review the ideas of the participants and be further motivated toward incorporating computer related mathematics and concepts into their classes.

Boulder, Colorado
August, 1967

C. C. Feng

TABLE OF CONTENTS

Page

PROJECT PERSONNEL ii

TEACHER PARTICIPANTS iii

STUDENT PARTICIPANTS iv

FOREWORD v

CALCULATING THE REAL ZEROS OF A REAL, POLYNOMIAL FUNCTION 1
R. L. Bassett, C. E. Friend and D. C. Lohrentz

USE OF COMPUTERS IN INSTRUCTION OF BASIC STATISTICS 17
Mildred J. Beavers

USING THE HARDY-WEINBERG LAW ON A COMPUTER 29
Robert L. Brown

PYTHAGOREAN TRIADS 43
Dewayne N. Burgdorff and Jay E. Niebur

NUMERICAL ANALYSIS AND DEMONSTRATIONS FOR P.S.S.C. LABORATORY EXPERIMENTS 54
Wayne B. Daniels

SOME USES OF THE COMPUTER IN PHYSICS TEACHING 62
William E. Einert and Hepburn Ingham

FACTORS G.C.F. PRIME FACTORIZATIONS 70
Sandra W. Epstein and Cheryl Gaskell

DATA EVALUATION FOR THE "CORRECT ANSWER" FOR JUNIOR HIGH SCHOOL STUDENTS 81
Margaret Ann Grill

INTRODUCTION TO FLOW CHARTING 94
John G. Groninger and Terry E. Shoemaker

USING THE COMPUTER FOR PERMUTATIONS AND COMBINATIONS 100
Neil O. Haflich and Ellis R. Mercer

THE SOLUTION OF A TYPICAL PROBLEM CONCERNING MOTION OF OBJECTS IN A GRAVITATIONAL FIELD 114
Marion E. Hakewessel

THE USE OF THE COMPUTER IN FINDING THE APPROXIMATE VALUE OF PI 118
Harry A. Hurley

Page

CONSTRUCTING FLOW CHARTS AND WRITING PROGRAMS FOR SIMPLE PROBLEMS 122
Dan H. Jarrell and Royce D. Jackson

THE INTERSECTION OF CIRCLES 140
Nick R. Kalianov and Pearl Price

FEEDBACK AROUND AN INTEGRATOR 146
Dean C. Larsen

A SHORT DESCRIPTION OF A NARRATED SERIES OF SLIDES WHICH PRESENT TWO INTRODUCTORY COMPUTER LESSONS FOR JUNIOR HIGH SCHOOL 156
D. Marcotte, Jr., K. W. Mahan and D. Colvin

A COMPUTER ORIENTED APPROACH TO LINEAR EQUATIONS IN ONE UNKNOWN 157
Robert A. Norton and Roe E. Willis

PRIME NUMBERS AND "BASIC" COMPUTER LANGUAGE 182
Raymond A. Platt and Kenneth A. Marine

AN APPROACH TO THE INTRODUCTION OF COMPUTER PROGRAMMING IN JUNIOR HIGH SCHOOL MATHEMATICS 192
Helen E. Schweizer

COMPUTER-ORIENTED SOLUTION OF A BALLISTIC PROBLEM 200
Billy L. Tipton and Marvin L. Miller

A COMPUTERIZED APPROACH TO GRAPHING PARABOLAS 204
Lavere C. Wilson

APPLYING THE COMPUTER TO A UNIT ON CIRCLES IN GEOMETRY 216
Karen L. Winquist

CALCULATING THE REAL ZEROS OF A REAL, POLYNOMIAL FUNCTION

By Robert L. Bassett, Carol E. Friend, and Donald C. Lohrentz

To find the real zeros of an Nth degree function there are five parts to this project:

I. Finding the integer roots of a function.
II. Searching for an interval where a root occurs.
III. Finding the real zeros of a function, half interval method.
IV. Finding the real zeros of a function, false position method.
V. Finding the real zeros of a function, Newton's method.

Any one of the five parts may be used, with only a minimum of programming experience.

Part I Finding the Integer Zeros of an Nth Degree Function

Part I could be used as introduction to the other parts, since the second year algebra student is familiar with this process. Parts III, IV and V are more difficult mathematically, therefore graphs are used to explain them. Part I of this project will assume that the reader has already covered the following topics:

A. The remainder theorem: The remainder when any polynomial $P(x)$ is divided by $(x-a)$ is $P(a)$.

B. The factor theorem: The factor theorem and its converse; $(x-a)$ is a factor of $p(x)$ if and only if $P(x)=0$ when $x=a$.

C. The fundamental theorem of algebra: Every complex polynomial equation of degree n has at least one complex number as a root.

Let's assume that an Nth degree polynomial is given by:

$f(x) = a_1x^n + a_2x^{n-1} + a_3x^{n-2} + \cdot \cdot \cdot + a_nx + a_{n+1}$

Problem: to find the integer values of x that make $f(x)=0$ or to find $\{x \epsilon I \mid a_1x^n + a_2x^{n-1} + a_3x^{n-2} + \cdot \cdot \cdot + a_nx + a_{n+1} = 0\}$

First set $f(x)=0$ and factor out an x from the first n terms. We then have: $x(a_1x^{n-1} + a_2x^{n-3} + a_3x^{n-4} + \cdot \cdot \cdot + a_n) + a_{n+1} = 0$

$x(a_1x^{n-1} + a_2x^{n-2} + a_3x^{n-4} + \cdot \cdot \cdot + a_n) = -(a_{n+1})$. If the integer truth set of $f(x) = 0$ is not empty then there exists an x or $(a_1x^{n-1} + a_2x^{n-2} + a_3x^{n-3} + \cdot \cdot \cdot + a_n)$ which is an integer factor of $-(a_{n+1})$. Therefore all of the integers that belong to the truth set would be factors of $-(a_{n+1})$.

Example $f(x) = x^3 - 2x^2 - 11x + 12$

Find $\{x \epsilon I \mid x^3 - 2x^2 - 11x + 12 = 0\}$

Therefore all members of the truth set would have to be factors of -12. Hence the only possible elements of the truth set are +1,+2,+3,+4,+6,+12,-1,-2,-3, -4,-6,-12. Using the remainder theorem, factor theorem and synthetic division, we would find that 1,4, and -3 are the only integers that satisfy $x^3 - 2x^2 - 11x + 12 = 0$. In other words the truth set is {1,4,-3}.

Checking each one of the possible roots is very time consuming, especially if it is done by hand computation. The next problem is how do you tell the computer to do all the time consuming computations? To keep the program simple the writer assumes that the reader can factor the constant term in f(x). Furthermore if the a_{n+1} term equals zero then zero is a root and any other integer root would be a factor of a_n. On the following page, Fig. 1 is a flow chart telling the computer how to do these computations.

PROGRAM

```
10 READ N
20 LET I=1
30 READ A(I)
40 LET I=I+1
50 IF I<=N+1 THEN 30
60 READ X
70 LET F=0
80 LET I=1
90 LET F=F*X+A(I)
100 LET I=I+1
110 IF I <+N+1 THEN 90
120 IF F=0 THEN 150
130 PRINT X "IS NØT AN INTEGER RØØT ØF F(X)"
140 GØ TØ 60
150 PRINT X "IS AN INTEGER RØØT ØF F(X)"
160 GØ TØ 60
170 DATA
180 DATA
190 END
```

Remarks

N is the degree of F(X), and must be entered in Step 170 Data [The value of N]. In steps 20 through 50 the computer is reading A_1, A_2, A_3, A_4, · · · A_n, A_{n+1}. These values should be entered in step 170 Data N, A_1, A_2, · · · A_{n+1}. In step 60 the computer reads your first factor of A_{n+1}. All X's should be entered in 180 Data X_1, X_2, X_3 · · · In Steps 70 through 110, F(X) is being generated. 120 if F = 0 then X is a root. 120 if F ≠ 0 then X isn't a root.

170 Data [The value of N]. [The values of A_1, A_2 · · · A_{n+1}]. 180 Data [The values of all the X's]. After all the data in 170 and 180 is used the computer stops.

Using this program, to solve for the integer roots of $x^3 - 2x^2 - 11x + 12 = 0$, we get the following:

```
FIND I    11:46   PX THU 08/24/67
 1     IS AN INTEGER RØØT ØF F[X]
 2     IS NØT AN INTEGER RØØT ØF F[X]
 3     IS NØT AN INTEGER RØØT ØF F[X]
 4     IS AN INTEGER RØØT ØF F[X]
 6     IS NØT AN INTEGER RØØT ØF F[X]
 12    IS NØT AN INTEGER RØØT ØF F[X]
-1     IS NØT AN INTEGER RØØT ØF F[X]
-2     IS NØT AN INTEGER RØØT ØF F[X]
-3     IS AN INTEGER RØØT ØF F[X]
-4     IS NØT AN INTEGER RØØT ØF F[X]
-6     IS NØT AN INTEGER RØØT ØF F[X]
-12    IS NØT AN INTEGER RØØT ØF F[X]
ØUT ØF DATA IN  60
FIND I    11:45   PX THU 08/24/67
10 READ N
20 LET I=1
30 READ A(I)
40 LET I=I+1
```

```
50 IF I<=N+1 THEN 30
60 READ X
70 LET F=0
80 LET I=1
90 LET F=F*X+A(I)
100 LET I=I+1
110 IF I<=N+1 THEN 90
120 IF F=0 THEN 150
130 PRINT X"IS NØT AN INTEGER RØØT ØF F(X)"
140 GØ TØ 60
150 PRINT X"IS AN INTEGER RØØT ØF F(X)"
160 GØ TØ 60
170 DATA 3,1,-2,-11,12
180 DATA 1,2,3,4,6,12,-1,-2,-3,-4,-6,-12
190 END
```

NOTICE: 170 Data 3,1,-2,-11,12. $N=3, a_1=1, a_2=-2, a_3=-11, a_4=12$.

180 Data 1,2,3,4,6,12,-1,-2,-3,-4,-6,-12. These are the factors of -12.

Part II Searching

Part II may be used as a preliminary program to Parts III, IV, or V to locate the interval where a root exists.

A discussion of the properties and behavior of polynomial functions is an essential topic in the study of elementary functions. Quite often a major portion of this discussion is concerned with calculating the zeros of a real polynomial function P(X), or equivalently, solving the polynomial equation P(X)=0 for its roots. Polynomial equations of the first or second degree may be solved directly by utilizing, respectively, basic properties of the real numbers or the quadratic formula. However, the solution of equations whose degree is greater than two is not as straight-forward and requires more ingenuity on the part of the student. We will be restricted to a presentation of calculating the real zeros of a real, polynomial function whose degree is three or greater. We note that odd-degree real equations will always have at least one real root since complex roots occur as conjugate pairs.

A classroom presentation of this subject would be preceded by the following related sequence of topics: remainder theorem, factor theorem, fundamental theorem of algebra and synthetic division. With the above knowledge the student is actually capable of solving a higher degree polynomial equation. However the solution process is mainly one of trial and error. Thus a typical classroom presentation would now involve the selection and teaching of time-saving techniques. Although the application of these techniques does not necessarily result in an actual solution they are useful since they give the number and approximate locations of the real roots.

A) Descarte's Rule of Signs: i) The number of positive roots of a polynomial is equal to, or less than, by an even integer, the number of sign changes in the coefficients. ii) The number of negative roots of a polynomial

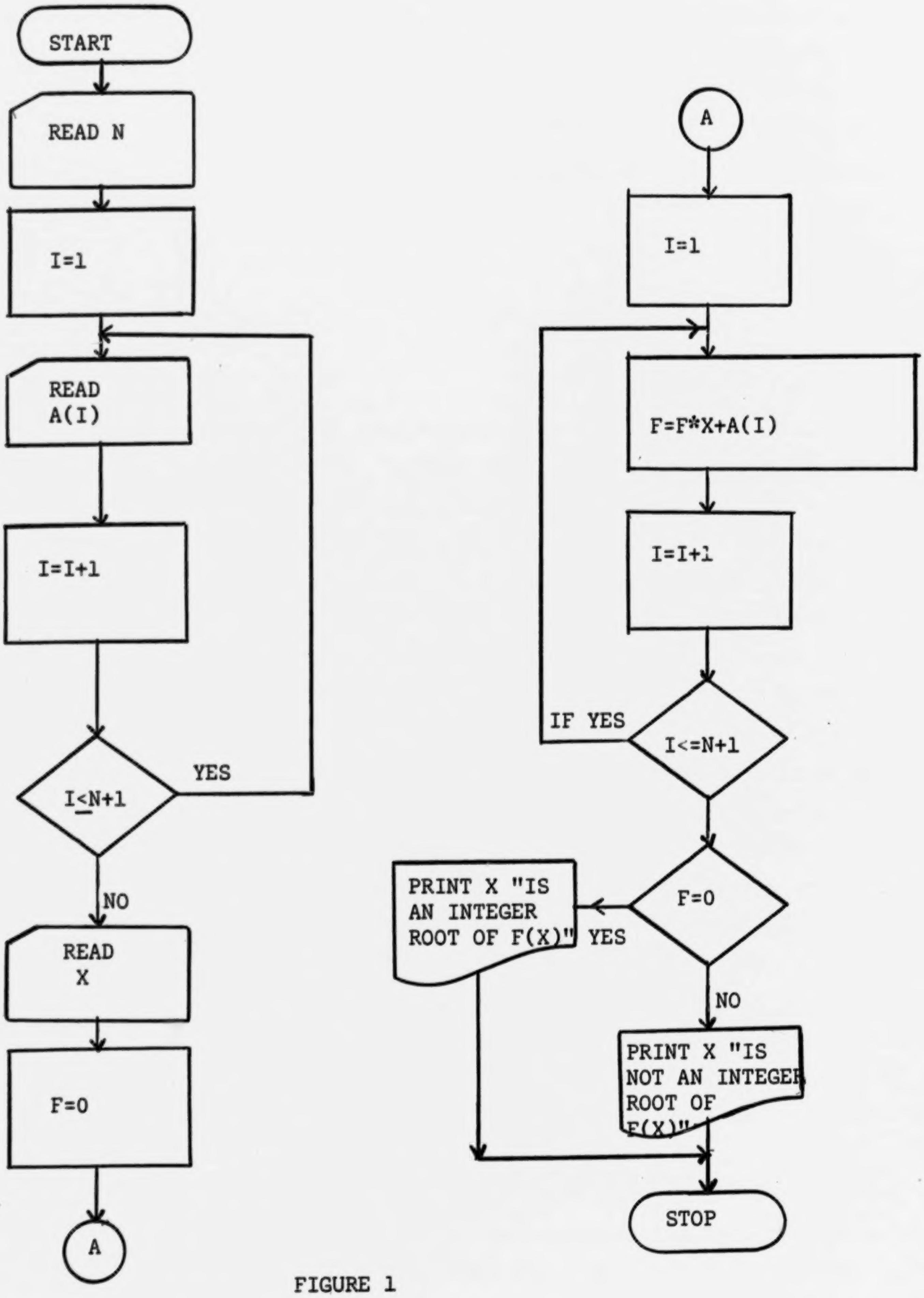

START
READ N
I=1
READ
A(I)
I=I+1
I<N+1
YES
NO
READ
X
F=0
A
A
I=1
F=F*X+A(I)
I=I+1
IF YES
I<=N+1
F=0
YES
PRINT X "IS
AN INTEGER
ROOT OF F(X)"
NO
PRINT X "IS
NOT AN INTEGER
ROOT OF
F(X)"
STOP

FIGURE 1

is equal to, or less than, by an even integer, the number of sign changes in the coefficients, if x is replaced by -x.

B) Newton's Relations: It may be shown by utilizing Newton's relations that if the roots are real, then the maximum value of the roots, in absolute value, is

$$X_{max} = \sqrt{\left(\frac{a_2}{a_1}\right)^2 - \frac{2a_3}{a_1}}$$

where $P(x) = a_1x^n + a_2x^{n-1} + \cdot\cdot\cdot + a_nx + a_{n+1}$.

C) Location Principle: If any real polynomial and real numbers a and b are given such that P(a) and P(b) have opposite signs, then there exists at least one real root of the equation P(x)=0 between a and b.

With the exception of those equations which have relatively small integers as their roots, the solution process is still tedious. The assigned problems must be few in number and relatively simple in solution. The question which often arises is, what course of action is taken if one has a large number of these equations to solve or the roots are not small integers? The answer is, use a computer.

The introduction of a computer approach to the solution of polynomial equations is particularly relevant at this point since it emphasizes the main advantage of a computer----it is a timesaving device. In the remainder of this paper we will present four programs concerned with the general solution of polynomial equations. The first will be a program for searching. That is, we will direct the computer to locate approximate values of the roots by searching for numbers a and b such that P(a) and P(b) have opposite signs. Secondly, we will present three programs which utilize these results to actually solve the equation for its roots.

In searching, the general procedure will be to first have the computer calculate the upper bound, R for the absolute value of the real roots and then to calculate function values at equally-spaced intervals from -R to R. The size of the increment, D is arbitrary. The computer will compare successive function values for opposite signs, and when they are found will print out x_1, $P(x_1)$, x_2, $P(x_2)$. A flow chart, Figure 2, and general program with explanations appear on the following pages.

The searching program was run with the following data statement:

```
280 DATA 3,20,24,-38,-65,25.
```

That is, we were searching for approximate loactions of the roots of $P(x) = 24x^3 - 38x^2 - 65x + 25$. The following results were obtained:

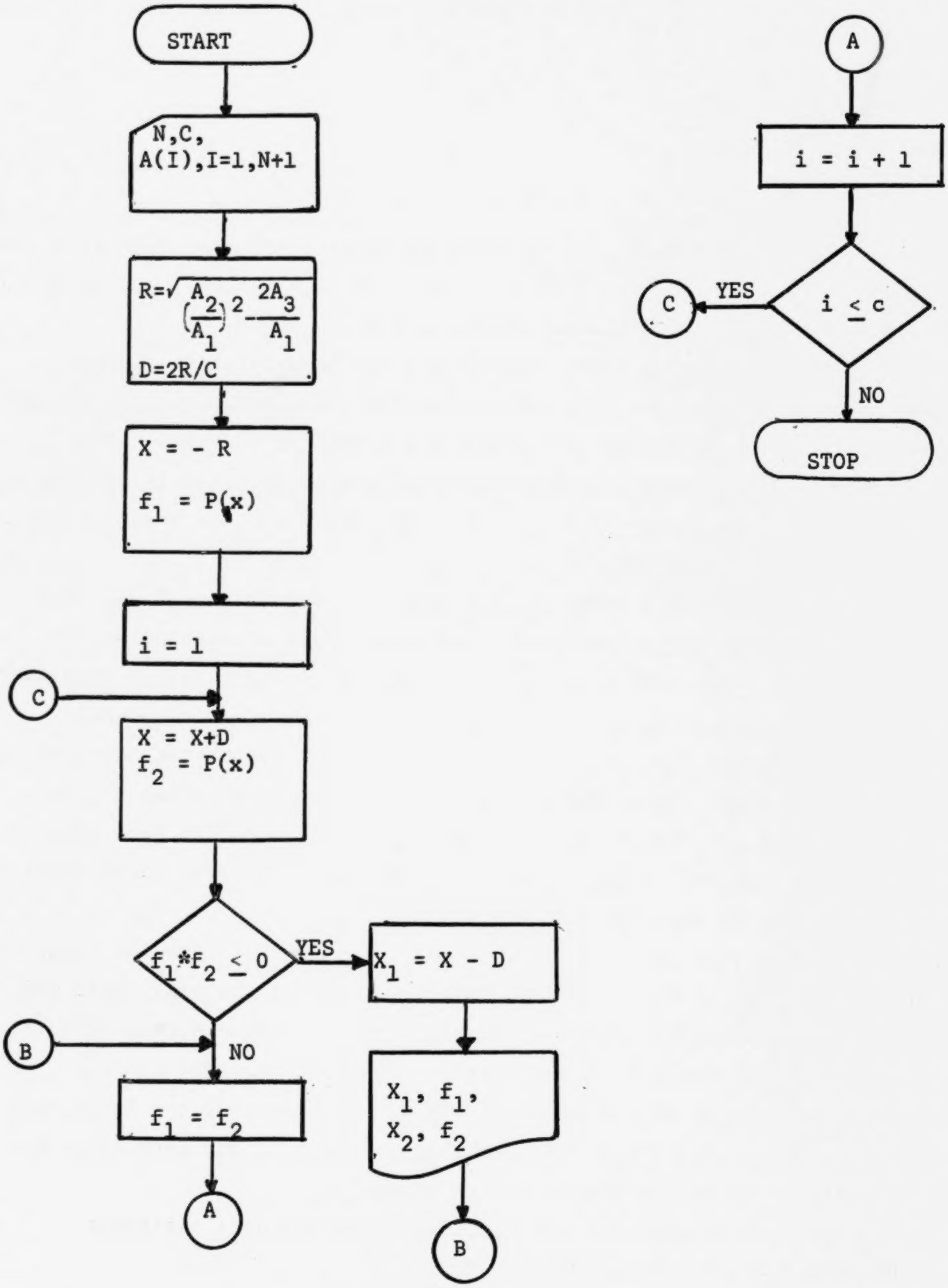

START
N,C,
A(I),I=1,N+1
$R=\sqrt{(\frac{A_2}{A_1})^2-\frac{2A_3}{A_1}}$
D=2R/C
X = - R
f_1 = P(x)
i = 1
C
X = X+D
f_2 = P(x)
f_1*f_2 ≤ 0
YES
X_1 = X - D
X_1, f_1,
X_2, f_2
B
NO
f_1 = f_2
A
i = i + 1
i ≤ c
YES
NO
STOP

FIGURE 2

```
-1.40745   P[X1] = -25.7027    -1.12596    P[X] =  15.7525

 .281489   P[X1] =   4.22754     .562978   P[X] = -19.3551

 2.25191   P[X1] = -40.0038     2.5334     P[X] =   6.67269
```

```
PROGRAM FOR SEARCHING                                            EXPLANATION

10 READ N,C---------------N = degree, C = number of equal increments
20 FØR I=1 TØ N+1
30 READ A(I)--------------coefficients of a particular polynomial
40 NEXT I
50 LET R=SQR((A2)/A(1))*(A(2)/A(1))-2*A(3)/A(1))----upper bound for roots
60 PRINT "MAXIMUM VALUE=" R
70 LET D=2*R/C------------calculate size of increment
80 LET X=-R---------------set x equal to lower bound
90 GØSUB 190--------------calculate P(x)
100 LET F1=P--------------set F1 equal to P(x)
110 FØR J=1 TØ C----------beginning of loop
120 LET X=X+D-------------increments x
130 GØSUB 190-------------calculate P(x+D)
140 LET F2=P--------------set F2 equal to P(x+D)
150 IF F1*F2<=0 THEN 240--if F1 and F2 have opposite signs then go to print
160 LET F1=F2                                                  statement
170 NEXT J----------------end of loop-returns to line 110
180 GØ TØ 290-------------have ended search for roots between -R and R
190 LET P=0
200 FØR K=1 TØ N+1--------lines 190 to 220 evaluate  the polynomial at the
210 LET P=P*X+A(K)                                last given value of x.
220 NEXT K
230 RETURN
240 LET X1=X-D------------calculates X1 for output purposes
250 PRINT X1,"P(X1)="F1,X,"P(X)="F2,
260 PRINT
270 GØ TØ 160-------------return to main loop and continue searching
280 DATA
290 GØ TØ 10--------------return to beginning if searching for roots of more
300 END                                                 than one polynomial
```

We may use results of this type in succeeding programs to actually calculate the roots.

Several facts should be noted at this point.

i) This program will give approximate locations for the real roots only.

ii) We can only assert that there is at least one root between the printed x values. There may be more.

iii) If the results of running the program are not completely successful, this may be remedied by running the program again and using a larger number of equal intervals. For instance, in the above example suppose that we had located only two roots. If we suspect that the third root is real we could substitute for line 280 the following:

```
280 DATA 3,50,24,-38,-65,25
```

and run the program again. With this change the computer will calculate function values at 50, instead of 20, equally-spaced points in the interval

-2.81489 to 2.81489. The greater the number of intervals the more refined the search.

iv) Another remedy is to increase the value of R. This may be necessary if any of the roots are complex since Newton's relations apply only if all the roots are real.

Part III Finding the Real Roots of any Polynomial, $f(x) = A_1x^n + A_2x^{n-1} + \cdot\cdot\cdot + A_nx + A_{n+1}$, By the Half Interval Method

A. Basic Idea: Two values, x_1 and x_2 are selected and tried. If it is found that $f(x_1)$ and $f(x_2)$ have opposite signs, then a real root exists for some value of x between x_1 and x_2. The new x will equal $(x_1 + x_2)/2$, see Fig. 3. This new x will replace either x_1 or x_2 and the process will continue until the root is found.

B. Running the Program: After the pre-punched program has been entered into the computer, change the three data statements (line numbers 500, 510, 520) to fit your program.

Example: 500 DATA 3 (This is the value for the order of the polynomial)
510 DATA 3,-2,4,3 (These are the values of the coefficients: A_1, A_2, A_3, A_4)
520 DATA -1,0 (These are the guesses for x_1 and x_2; the computer will try to find a root somewhere between these values)

Run the program.

Note - The computer will determine ONE of the following conditions for a given set of data:

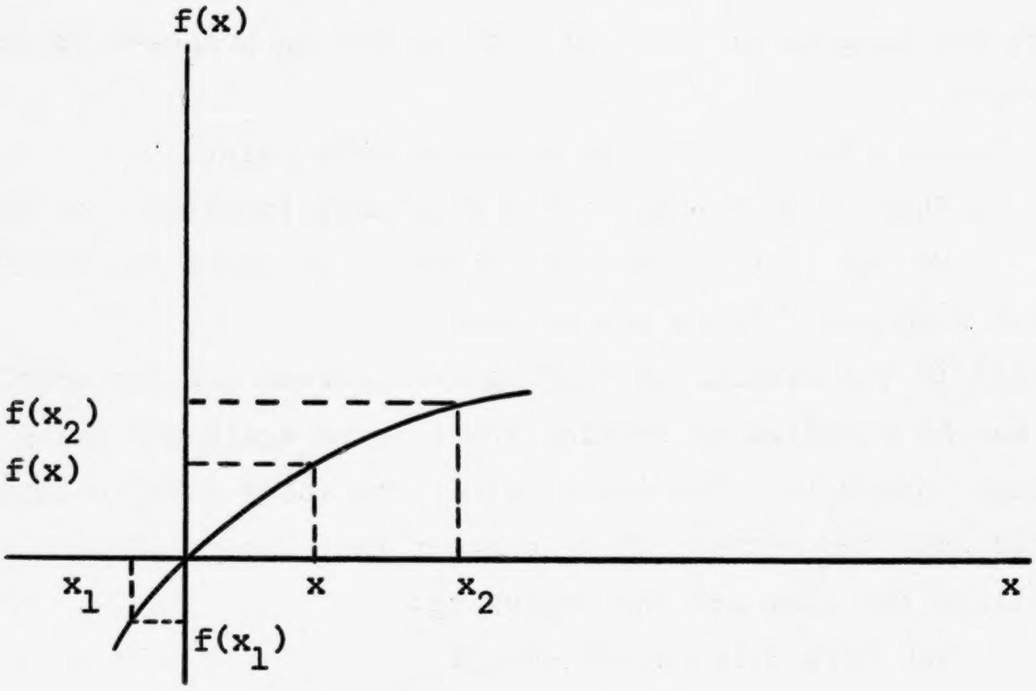


FIGURE 3

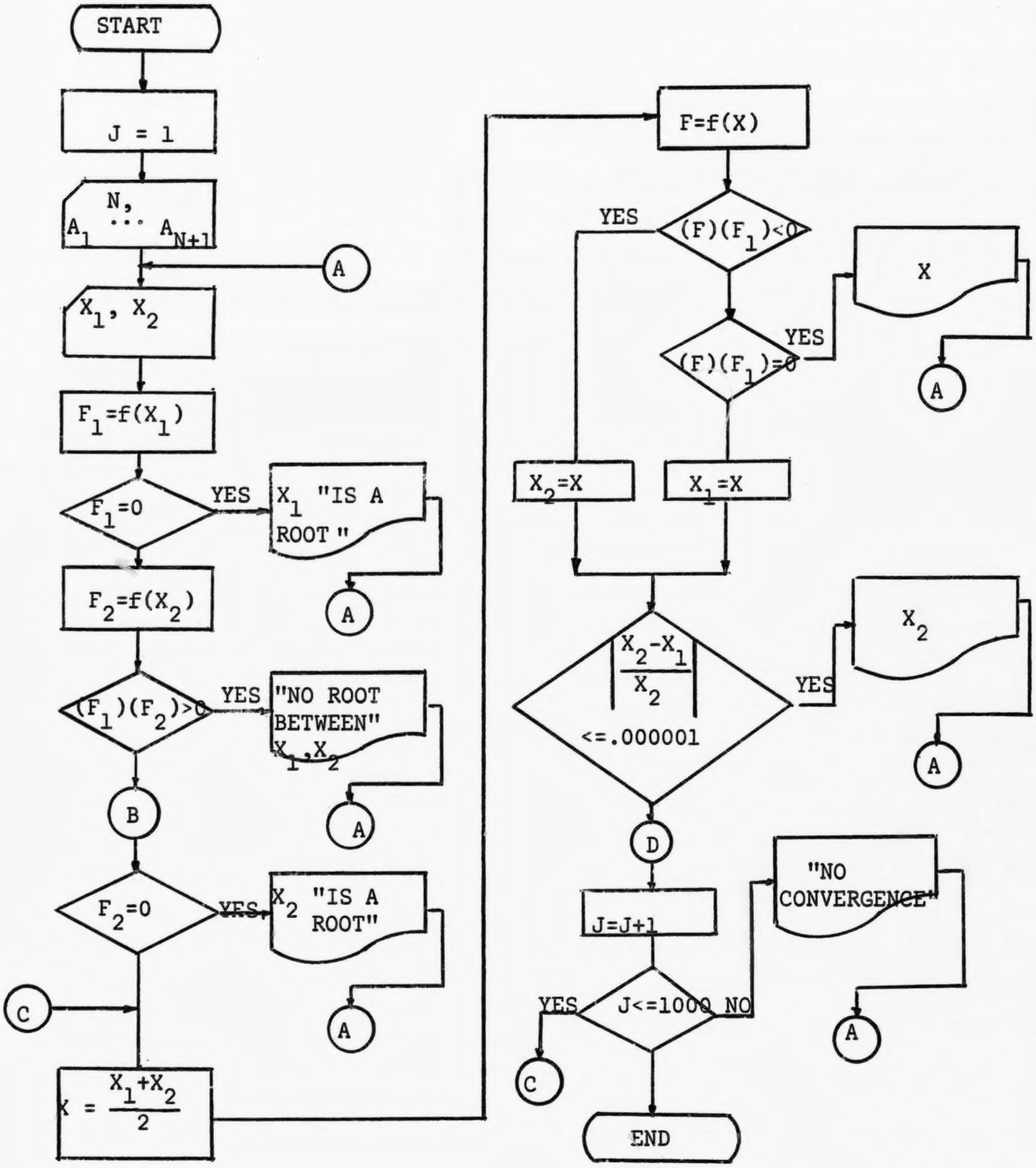

START
$J = 1$
N, $A_1 \cdots A_{N+1}$
A
X_1, X_2
$F_1=f(X_1)$
$F_1=0$
YES
X_1 "IS A ROOT"
A
$F_2=f(X_2)$
$(F_1)(F_2)>0$
YES
"NO ROOT BETWEEN" X_1, X_2
A
B
$F_2=0$
YES
X_2 "IS A ROOT"
A
C
$X = \frac{X_1+X_2}{2}$
$F=f(X)$
YES
$(F)(F_1)<0$
X
$(F)(F_1)=0$
YES
A
$X_2=X$
$X_1=X$
$\left|\frac{X_2-X_1}{X_2}\right|$ <=.000001
YES
X_2
A
D
$J=J+1$
"NO CONVERGENCE"
YES
$J<=1000$
NO
A
C
END

FIGURE 4

1. x_1 is a root.
2. x_2 is a root.
3. There is no real root between x_1 and x_2.
4. The exact value of a root.
5. The value of a root to an accuracy of ± 0.000001.
6. If one of the above conditions is not met after 1000 divisions, the computer will print "NO CONVERGENCE".

C. The program written in <u>Basic</u> Language (See Flow Chart, Figure 4)

```
PROGRAM                                 COMMENTS

10 LET J=1--------------See steps 270,280
20 READ N---------------Value of the order of the polynomial.
30 FOR K=1 TO N+1-------Steps 30,40,50 get the coefficients into the computer.
40 READ A(K)
50 NEXT K
60 READ X1, X2----------Two guesses for x; the computer will determine if
                          there is a root between them.
70 LET F1=0-------------Steps 70,80,90,100 compute the value of the polynomial
80 FOR K=1 TO N+1         at x1 using the form:  f(x1) = [[[[A1x + A2]x + A3]x
90 LET F1=F1*X1+A(K)
100 NEXT K                + · · · + An]x + An+1]
110 IF F1=0 THEN 310----A check to determine if x1 is a root.
120 LET F2=0------------Steps 120,130,140,150 compute the value of the
130 FOR K=1 TO N+1        polynomial at x2 (same form as steps 70-100).
140 LET F2=F2*X2+A(K)
150 NEXT K
160 IF F1*F2>0 THEN 330-If the product of f(x1) and f(x2) is a positive number,
                          then no root can be found between them.
170 IF F2=0 THEN 350----A check to determine if x2 is a root.
180 LET X=(X1+X2)/2-----A new value for x halfway between x1 and x2.
190 LET F=0-------------Steps 190,200,210,220 compute the value of the
200 FOR K=1 TO N+1        polynomial at the new x (same form as steps 70-100).
210 LET F=F*X+A(K)
220 NEXT K
230 IF F*F1<0 THEN 390--If the product of f(x) and f(x1) is negative, a root
                          lies between them.
240 IF F*F1=0 THEN 370--If f(x) is zero then the product will be zero and x
                          is a root.
250 LET X1=X------------Replace x1 by x because the product of f(x) and f(x1)
                          is positive.
260 IF ABS((X2-X1)/X2)< If interval is within 0.000001, then x2 is printed.
    = .000001 THEN 410
270 LET J=J+1-----------Steps 270,280,290.  After 1000 tries without success
280 IF J<=1000 THEN 180   in 260, "NO CONVERGENCE" is printed.
290 PRINT "NO CONVERGENCE"
300 GO TO 60------------(Also 320,340,360,380,400,420) After solving for a
310 PRINT X1"IS A ROOT"   given x1 and x2 the computer goes to step 60 and
320 GO TO 60              begins again with a new x1 and x2 read from line 520.
330 PRINT "NO ROOT BETWEEN"
    X1 "AND" X2
340 GO TO 60
350 PRINT X2 "IS A ROOT"
360 GO TO 60
370 PRINT X
380 GO TO 60
```

```
390 LET X2=X---------------x2 is replaced by x.
400 GO TO 260
410 PRINT X2
420 GO TO 60
500 DATA 3-----------------Data for step 20.
510 DATA 3,-2,4,3----------Data for steps 30,40,50.
520 DATA -1,0--------------Data for step 60.
599 END
```

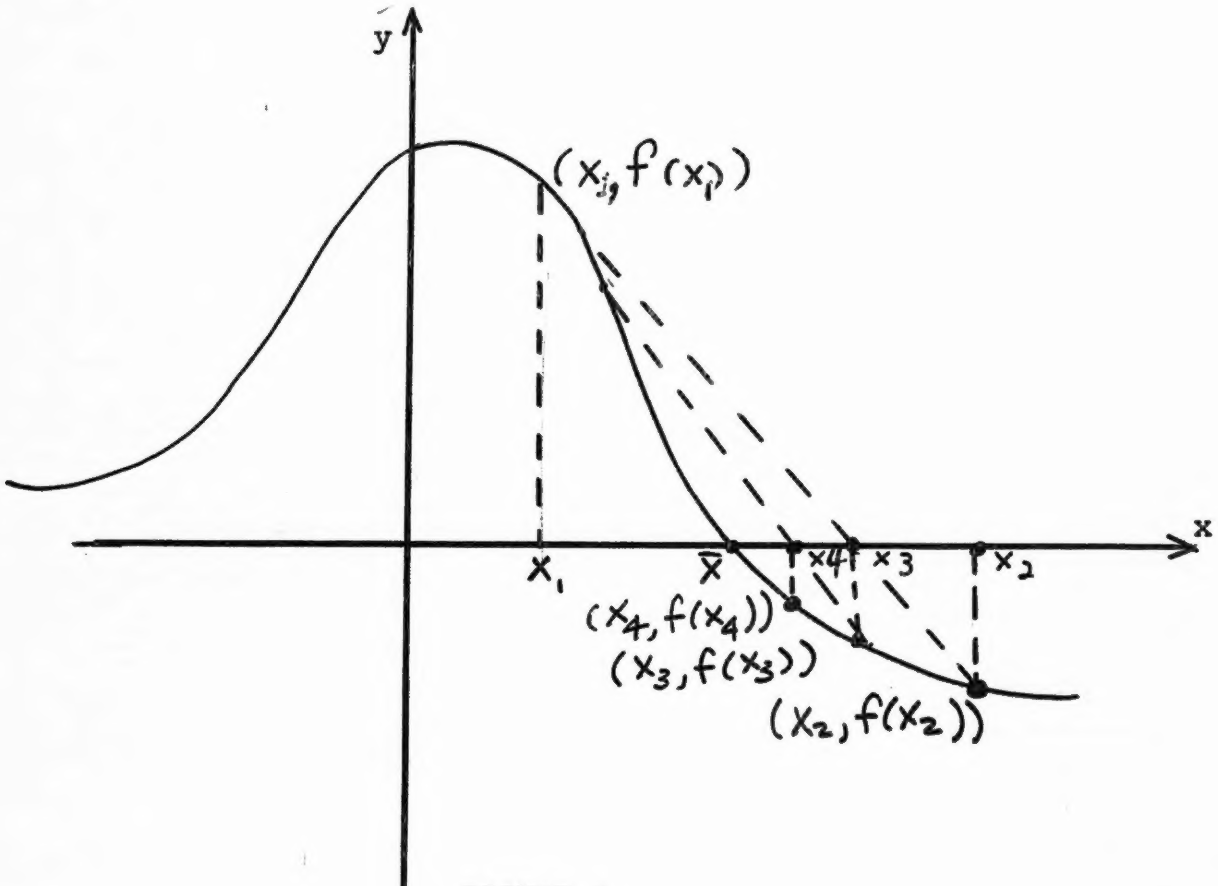


FIGURE 5

Part IV Method of False Position

The method of false position utilizes the results of the searching program to calculate the roots of a polynomial equation. Figure 5 gives a geometrical interpretation of this method. (x_1, f_1) and (x_2, f_2) are points on the graph of the polynomial function such that f_1 and f_2 have opposite signs. The secant determined by these two points is introduced. By a similar triangle argument we may solve for x_3, the point of intersection of the secant and the x-axis.

$$\frac{x_3 - x_1}{x_2 - x_1} = \frac{f_1}{f_1 - f_2} \qquad x_3 = \frac{x_1 * f_2 - x_2 * f_1}{f_2 - f_2}$$

Note that this equation will also work if $f_1 < f_2$. The only stipulation is that $x_1 < x_2$. In this particular instance since $f_1 * f_2 < 0$ we introduce the secant determined by (x_1, f_1) and (x_3, f_3), and procede to solve for x_4 in the same fashion as before. If $f_1 * f_2 > 0$ we would have introduced the secant determined by (x_2, f_2) and (x_3, f_3). We see that the points x_2, x_3, x_4, $\cdots$, etc. are converging on the value $\bar{x}$ which is a root of the polynomial.

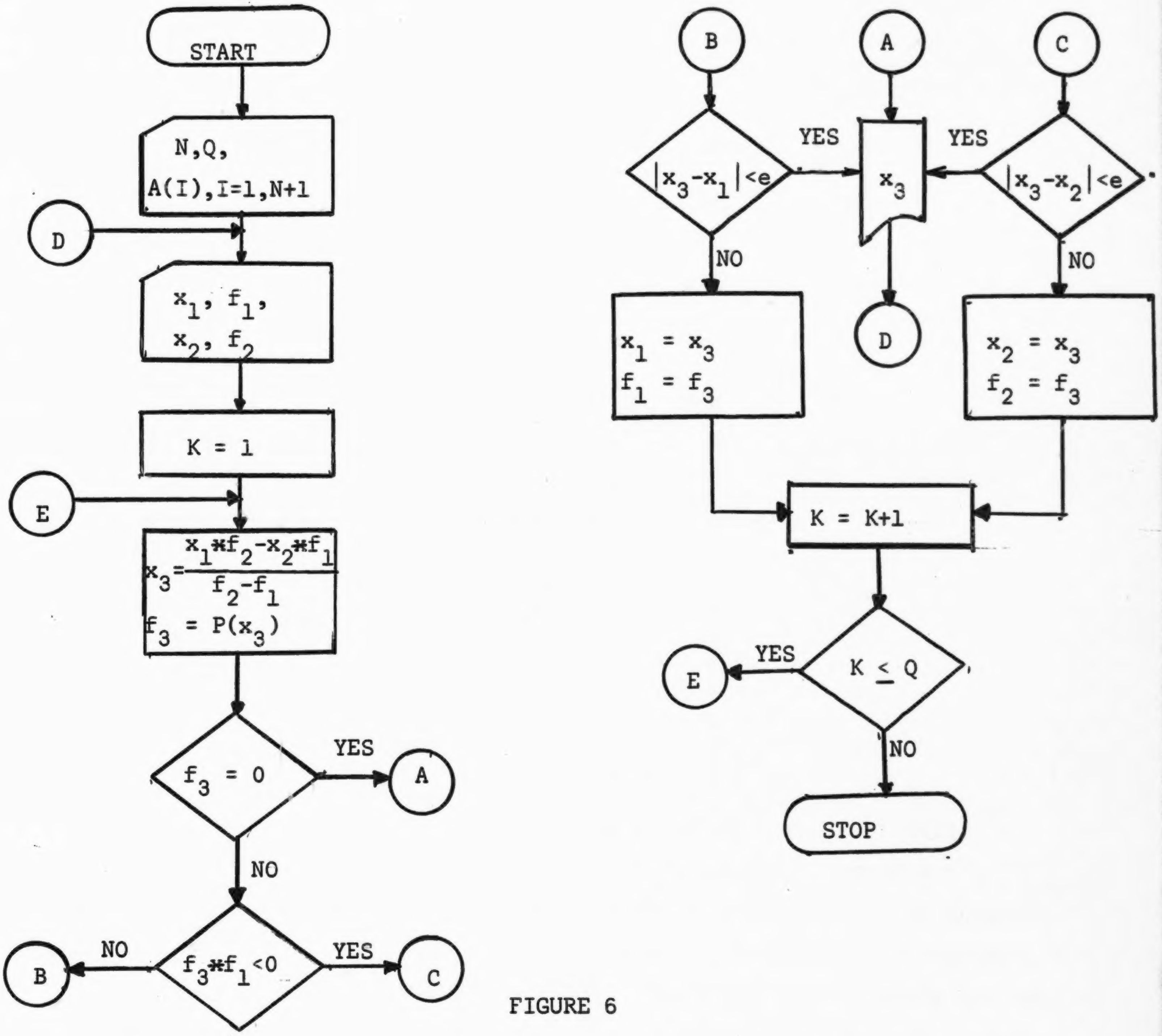

START
N,Q,
A(I),I=1,N+1
D
x_1, f_1,
x_2, f_2
K = 1
E
$x_3 = \frac{x_1*f_2-x_2*f_1}{f_2-f_1}$
$f_3 = P(x_3)$
$f_3 = 0$
YES
A
NO
NO
B
$f_3*f_1<0$
YES
C
B
A
C
$|x_3-x_1|<e$
YES
x_3
YES
$|x_3-x_2|<e$
NO
NO
$x_1 = x_3$
$f_1 = f_3$
D
$x_2 = x_3$
$f_2 = f_3$
K = K+1
YES
E
K ≤ Q
NO
STOP

FIGURE 6

A flow chart, Figure 6 above, and program with explanations follow on the next pages.

This program was run with the following data statements. We are using the results of the searching program.

```
250 DATA 3,1000,24,-38,-65,25,-1.40745,-25.7027
260 DATA -1.12596,15.7525,.281489,4.22754,.562978
270 DATA -19.3551,2.25191,-40.0038,2.5334,6.67268
```

The following results were obtained:

```
-1.25
 .333333
2.5
```

Substitution into $P(x) = 24x^3 - 38x^2 - 65X + 25$ reveals that these are the exact values of the zeros.

The following items about the program should be noted:

i) The maximum number of iterations, Q, is introduced to prevent an infinite loop.

ii) Lines 150 and 190 calculate the difference between two successive approximations of the root.

```
PROGRAM FOR METHOD OF FALSE POSITION             EXPLANATION

10 READ N,Q-----------------------N=degree Q=counter
20 FØR I=1 TO N+1
30 READ A(I)----------------------Coefficients of the polynomial
40 NEXT I
50 READ X1.F1,X2,F2---------------Values obtained from searching
60 FØR K=1 TØ Q-------------------Beginning of loop
70 LET X3=(X1*F2-F2*X1)/(F2-F1)---Calculate X3
80 LET P=0------------------------Lines 80 to 110 calculate P(x3)
90 FØR J=1 TØ N+1
100 LET P=P*X3 + A(J)
110 NEXT J
120 LET F3=P----------------------Set F3=P(x3)
130 IF F3=0 THEN 230--------------IF P(x3)=0 Then print
140 IF F3*F1<0 THEN 190-----------If F3 and F1 have opposite signs then go to 190
150 IF ABS(X3-X1)<=10E-6 THEN 230-Calculate difference between successive
160 LET X1=X3                                                approximations
170 LET F1=F3---------------------Lines 160 and 170 reassign values to X1 and F1
180 GØ TØ 220                       since F1 and F3 have the same sign.
190 IF ABS(X3-X2)<=10E-6 THEN 230-Same as 150
200 LET X2=X3---------------------Lines 200 and 210 reassign values to X2 and F2
210 LET F2=F3
220 NEXT K------------------------End of loop
230 PRINT X3
240 GØ TØ 50---------------------Go to line 50 for calculation of next root
250 DATA
280 END
```

If the difference is sufficiently small the process is stopped.

iii) This method will not work if a given root is of even multiplicity. In this case the graph of the function is tangent to the x-axis.

Part V Finding the Real Roots of Any Polynomial, $f(x)=A_1x^n + A_2x^{n-1} + \cdots + A_nx + A_{n+1}$, by Newton's Method.

A. Basic Idea: A value for x_1 is tried near the value of the desired root, and $f(x_1)$ is computed. The intersection of the tangent to the curve at $(x_1,f(x_1))$ and the x axis gives the new value of x to be computed (see Fig. 7). This process is repeated until the root is found or until the desired closeness is attained.

B. Running the Program: After the pre-punched tape has been entered into the computer, change the three data statements (line numbers 300,310, 320) to fit your program.

Example: 300 DATA 3 (This is the value for the highest power of x).

310 DATA 3,-2,4,3 (These are the values of the coefficients: A_1, A_2, A_3, A_4)

320 DATA 0 (This is the first trial for x_1; it should be chosen near the root).

Run the program.

Note - The computer will determine ONE of the following conditions for a given value of x_1:

1. The value of a root to an accuracy of ± 0.000001.
2. If the above accuracy is not attained after 1000 cycles, the computer will print NO CONVERGENCE.

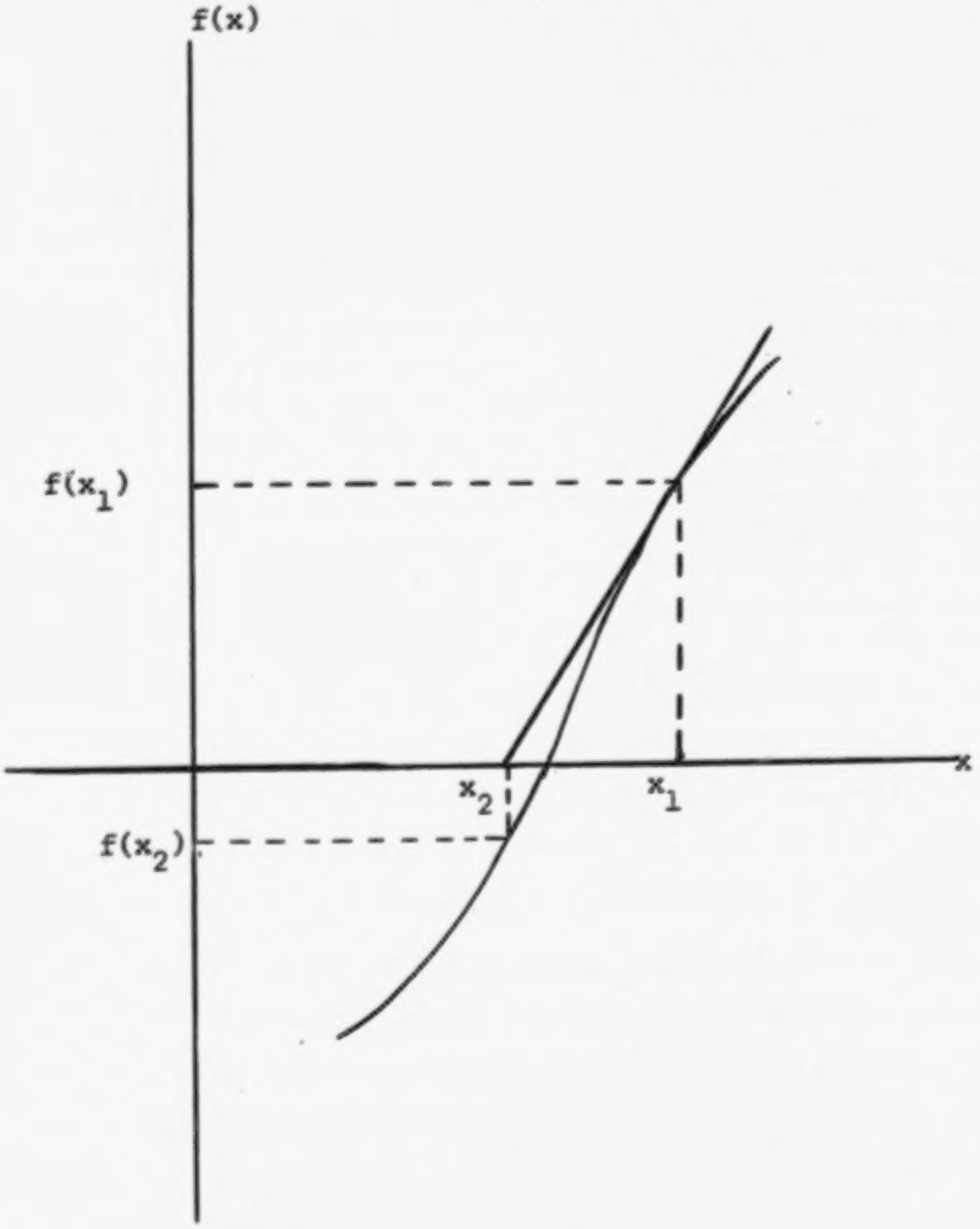


FIGURE 7

C. Derivation of the Function: Definition of the slope m of a tangent line is:

$m = f(x_1)/(x_1-x_2)$

from which we get:

$x_2 = x_1 - f(x_1)/m$

from calculus, the slope m is the derivative of the function at point x_1:

$m = f'(x_1)$

so, Newton's equation for the second trial is:

$x_2=x_1-f(x_1)/f'(x_1)$

D. The Program Written in Basic Language (See Flow Chart, Fig. 8)

```
    PROGRAM                                COMMENTS

10 LET J=1------------------See 180,190.
20 READ N-------------------The largest exponent of x.
30 FOR K=1 TO N+1-----------Coefficients of the polynomial are entered in
40 READ A(K)                   steps 30,40,50.
50 NEXT K
60 READ X1------------------A number near the root should be entered.
70 LET P=0------------------Steps 70,80,90,100. f(x1) is computed using the
80 FOR K=1 TO N+1              form f(x1) = [[[[A1x+A2]x + A3]x + · · · +
90 LET P=P*X1+A(K)
100 NEXT K                     An]x + An+1]
110 LET Q=A(1)*N------------Steps 110,120,130,140. f'(x1) is computed using
120 FOR K=2 TO N               the form f'(x1) = [[[[A1nx+A2(n-1)]x + A3(n-2)]
130 LET Q=Q*X1+(N-K+1)*A(K)    + · · · + An-1(2)]x + An]; where D_x x^n = nx^(n-1) .
140 NEXT K
150 LET X2=X1-P/Q-----------x2 = x1 - f(x1)/f'(x1); see Part C, Derivation of
                               the Function.
160 IF ABS((X2-X1)/X2)<  ---If the interval is within ± 0.000001, then x2 is
    =.000001 THEN 220          printed.
170 LET X1=X2---------------x1 is replaced by x2.
180 LET J=J+1---------------Steps 180,190,200. After 1000 tries without
190 IF J<=1000 THEN 70         success in step 160, "NO CONVERGENCE" is printed.
200 PRINT "NO CONVERGENCE"
210 GO TO 60----------------Steps 210,230. After solving for a given x1,
                               the computer goes to step 60 and begins again
220 PRINT X2                   with a new x1 read from line 320.
230 GO TO 60
300 DATA 3------------------The order of the polynomial.
310 DATA 3,-2,4,3-----------The values of the coefficients.
320 DATA 0------------------Your first trial for x1.
399 END
```

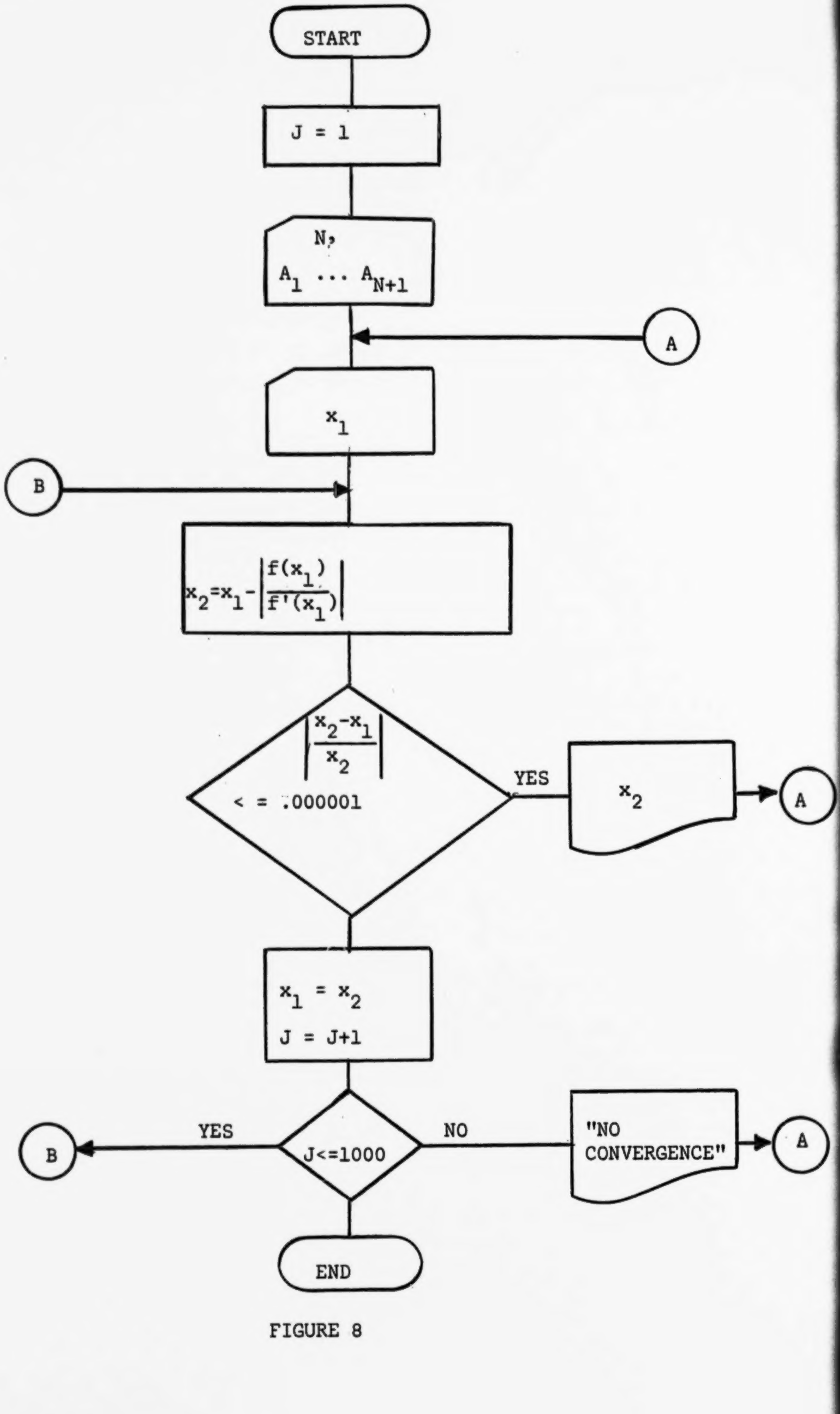

START
J = 1
N,
$A_1 \ldots A_{N+1}$
A
x_1
B
$x_2=x_1-\left|\frac{f(x_1)}{f'(x_1)}\right|$
$\left|\frac{x_2-x_1}{x_2}\right|$ < = .000001
YES
x_2
A
$x_1 = x_2$
J = J+1
B
YES
J<=1000
NO
"NO CONVERGENCE"
A
END

FIGURE 8

USE OF COMPUTERS IN INSTRUCTION OF BASIC STATISTICS

by Mildred J. Beavers

One mathematics class available to twelfth graders at Boulder High School is Senior Math. This class was organized to fill a need for additional math beyond one year of algebra and one year of geometry for the student who is not primarily interested in further study of mathematics or science.

The classes are composed of several segments. Most who take the course are college bound (95%) but will continue in other subject matter areas. Some take it to improve college board scores; a few, only because they need a credit. Included in the group are many who have had unpleasant experiences with mathematics earlier but realize that they should have greater knowledge for cultural as well as economic reasons

Senior Math serves as a review of axioms and basic theorems of both algebra and geometry. Methods of solutions for linear and quadratic equations are rediscovered. New topics include fractional exponents, arithmetic and geometric sequences and series, permutations, combinations, trigonometric functions, solution of triangles, simple identities, logic, and others. All are taught without the rigor that is demanded of math majors. Many concepts are treated intuitively; others with proof. The principle text book used in the course is Advanced High School Mathematics; Vannatta, Carnahan, Fawcett; Merrill, 1961.

This topic chosen for computer adaption is the one on descriptive statistics. Formulas are given based on simplified definitions. The assumption that students have acquired a beginning knowledge of "Basic Language" has been made. Therefore, the first task is to define terms: i.e., data, datum, statistics, descriptive statistics as different from inferential statistics.

Lesson 1 -- Arithmetic Mean

Definition -- Arithmetic mean -- a measure of central tendency -- the sum of all data divided by the number of elements in the data.

Examples

1) The number which represents the ultimate score which is their grade for the quarter or semester.

2) Average earnings per week based on a sum of 52 weeks' salary.

Definition -- Summation Symbol (This is the first time most of these students have seen the summation symbol.)

$$\sum_{i=1}^{n} X_i$$

Definition -- $\overline{X}$ is used to represent the arithmetic mean.

$$\overline{X} = \frac{1}{n} \sum_{i=1}^{n} X_i$$

The exercises on page 254, 1, 2, 3, 4, 5, would be assigned. These are artificial situations, overly simplified in nature.

Example 1. Find the mean of 87.2, 68.5, 74.8, 94.0, 82.2, 96.1.

The other 4 examples are about the same.

This is a matter of 10 minutes at the most in the classroom.

Now, in order to demonstrate the speed and efficiency of the computer, I would take the students through the flow chart and the program for the computer. These could be written on the board, or the overhead projector might be used. Under either set of circumstances, the two would be used side-by-side, explaining the flow chart first, and leading into the program.

Problem 4 of the set is good to demonstrate the program:

John's per cent grades on five tests were 84, 72, 91, 64, 83.

Find the mean.

After the explanation using Problem 4 and assuming that a computer were available, the class would then have a demonstration of their problems as solved by a computer. Since they have worked unrealistic examples, these examples should now be contrasted with an example of the magnitude they might find in the real world. In order to make the contrast, they would put through the computer the arithmetic mean of the automobiles per state in the United States. The numbers of automobiles are expressed as 1000

```
111 DATA 51,1349,72,635,643,8371,878,1250,203,209,2637,1612
112 DATA 1873,3485,1541,674,1646,289,642,198,275,2644,388,5282
113 DATA 1699,260,4392,1043,907,4326,361,896,279,1339,4393
114 DATA 405,136,1508,1310,554,1530,148
```

The arithmetic mean is 1468.69. This indicates 1,468,690 as the average number of cars per state in 1965. This took the computer less than 1 second to do.

The comparison of the two should begin to show a little of the power of the computer.

Lesson 2 -- Other Measures of Central Tendency

Definitions

1) Array -- the raw data is in no special arrangement. If these numbers are arranged into an ordered sequence, usually in order of increasing size, then this sequence is called an array.
2) Median -- the median is the mid-value of a set of data. If the number of data is odd, then the median is the mid-odd value.

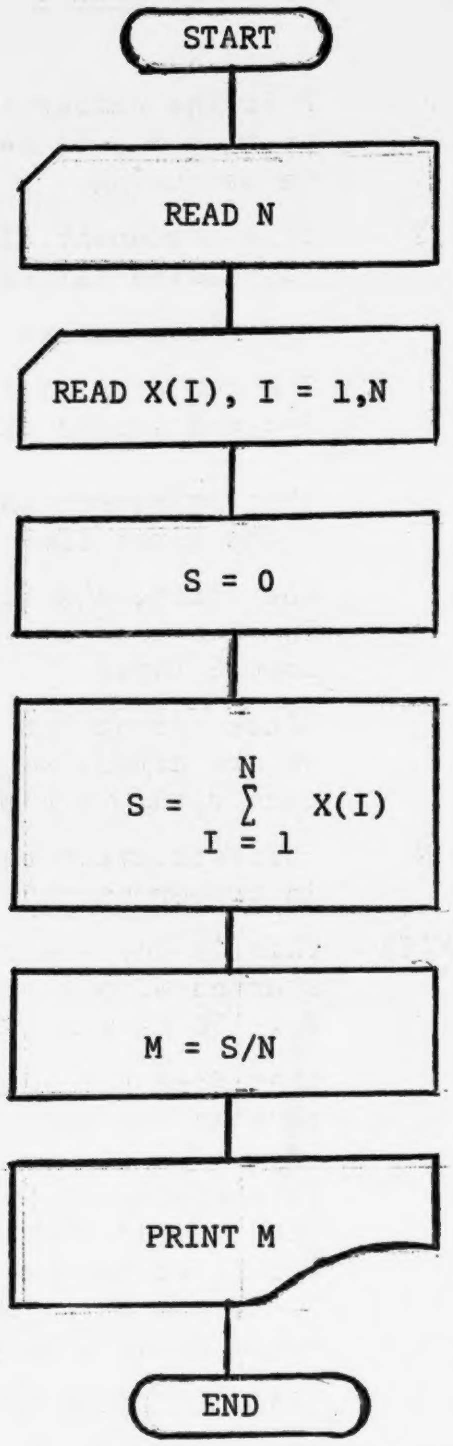

START
READ N
READ X(I), I = 1,N
S = 0
S = $\sum_{I=1}^{N}$ X(I)
M = S/N
PRINT M
END

Figure 1

Program 1

10 READ N	N is the number of elements in the data. In this problem N = 5, because there are 5 test scores that we are using.
20 FOR I = 1 TO N	This statement directs the X(I) in 30 to take on all successive values of X_i from i = 1 to 5. These X_i's are those values listed in the data.
30 READ X(I)	The computer will read 84 the first time, since X is 84; the second time through $X_i = X_2$, which is 72.
40 NEXT I	This statement says that the computer should use the X the first time it reads X(I); X_2, the second; X_3, the third, and so on until the list of data is used. In this problem the computer would go through this loop 5 times.
50 LET S = 0	Since the sum of the data is the first calculation we are after, we start with a sum of zero and add each datum to the previously calculated sum.
60 FOR I = 1 TO N	This statement tells the computer to go use the X's in proper sequence for the summing operation.
70 LET S = S + X(I)	This is key statement for the addition process. We started with a sum of zero in 50. 60 said read first X_i. 70 says add 84 (XI) to 0. This is the statement that does the summation operation, giving us partial sums as the computer uses successive X_i's.
80 NEXT I	This statement tells the computer to go back and read 72 and then in 70 add 72 to 84. The next time add 91 to the sum of 72 + 84, namely 156. New S in 70 is now 247. This is a partial sum. The computer keeps doing this until the last X_i, in this case X_5 (83), is added to the previous partial sum.
90 LET M = S/N	This command tells the computer to divide the sum by N(5).
100 PRINT "THE MEAN=" M	Here the statement "The mean is 78.8"
110 DATA 5,84,72, 91,64,83	
111 DATA	Other problems
112 DATA	Other problems
115 GØ TØ 10	If you wish to do several problems the computer will go back to 10, read new N, and new data. If you have several data lines for one stop program change 115 to "GØ TØ 20."
120 END	

Thus, if there are 9 numbers in the data, the 5th number would be the median. If there are an even number of data, then the median is the arithmetic mean of the middle two numbers. 12 numbers in an array would yield a median which would be the arithmetic mean of the 6th and 7th numbers.

3) Range -- the range is the difference between the largest and smallest numbers of the array.

The examples in the text are again trivial in nature. A quick calculation of Problems 1 and 2 will give enough practice to understand the principles involved. Problem 4 -- "Will the median be affected if the numbers at the extremes of an array are changed? Will the mean be affected by such changes? Can the mean and median ever coincide?" This problem lends itself to experimentation with the data. First the suggestion could be made to the class that they try adding data to the array in patterns; i.e., adding numbers at either end which were greater and lesser by the same value, adding a value to the upper end that is twice that of the one added to the lower end of the array. All sorts of arrangements of this nature can be explored and intelligent guesses recorded in a temporary place.

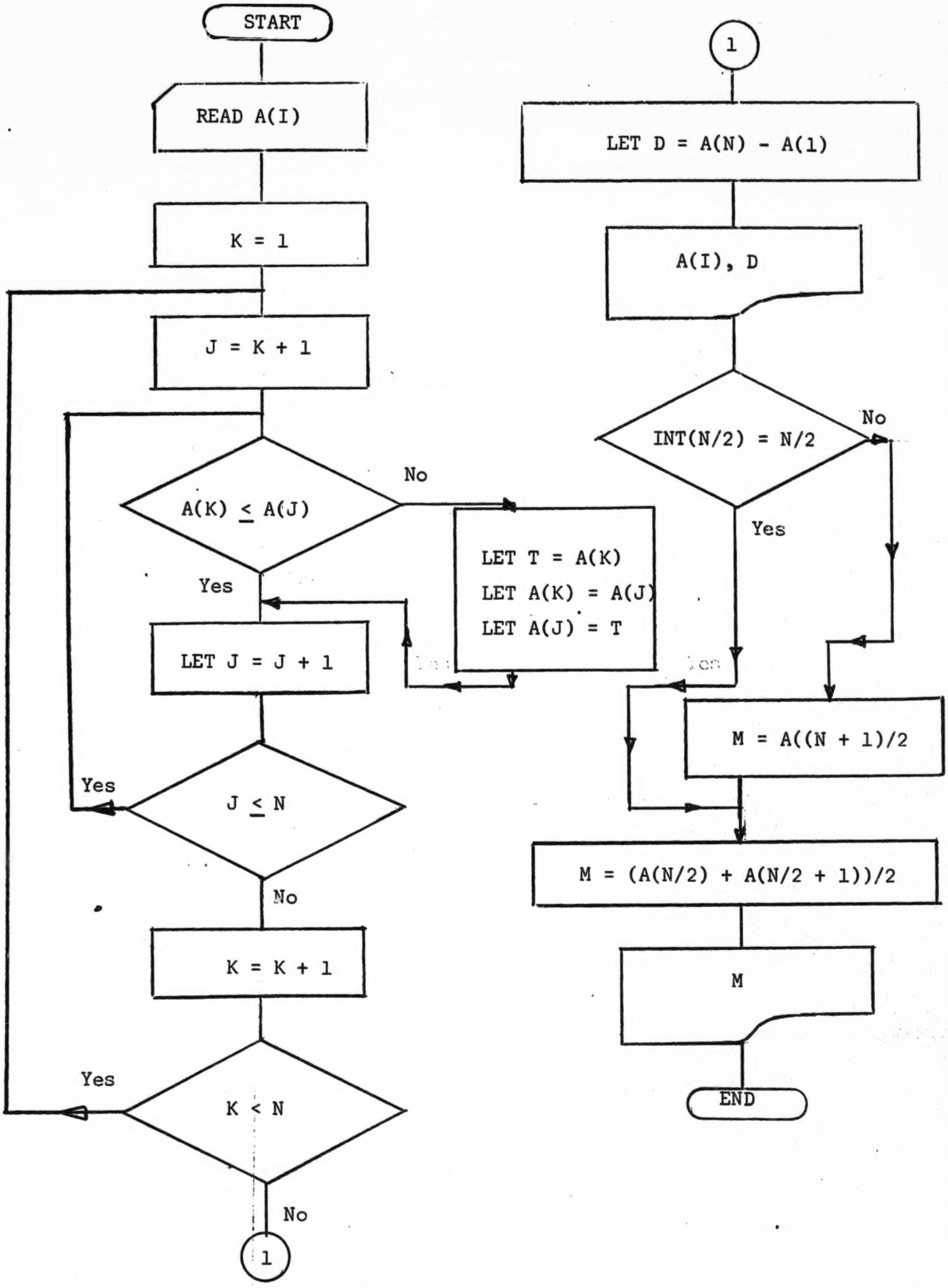
Then on the overhead projector or chalkboard, the flow chart and program could be written. In this lesson, instead of giving a complete explanation immediately, give the class a chance to recognize what they can in both. The recognition will be limited, but commands which are mathematical in nature as well as previously used statements can be noted. Then go through both flow chart and program, giving a detailed account of what is happening. Example 1 in the set of exercises will serve to illustrate.

Example 1 -- Find the median of $64, $82, $51, $90, $67, $71, $58, $94, $63. What is the range? (See Figure 2.)

Again, the problem involving the number of automobiles would serve to illustrate the speed and competency of the computer. The time that the problem took the computer to form an array and calculate range and median should be pointed out so that students are reminded of one of the chief assets of the computer. To emphasize this, the instructor might even time the students as they tried to do the same problem. Let them pick any part of the problem to do but use the same set of data. Start with a stop watch, and call time. Using the same data as in the previous lesson, the computer forms an array, calculates the range (8299) and the median (1043) in 6 seconds. Compare the

Program for Forming an Array, Range, and Median

10 READ N	N is number of elements in data.
20 FOR I=1 TØ N 30 READ A(I) 40 NEXT I	20-40 -- this subscripted variable technique was explained in previous lesson. It simply instructs the computer to read the data in sequence.
50 LET K=1 60 LET J=K+1	50-60 -- these two steps set up a routine, making J one unit more than K where J and K are both values of I.
70 IF A(K)<=A(J)THEN 110	Compares A(K) to A(J). If A(K) is less than or equal to A(J), then the numbers are in an order with the smaller one first, which is what the program is telling the computer to do -- arranged in ascending order. If A(K)≤A(J) then the computer goes to 110.
80 LET T=A(K) 90 LET A(K)=A(J) 100 LET A(J)=T	80-100 -- if A(K)>A(J) this means they are not in ascending order, and the order must be changed. To do this, the commands as they are written assign the value A(K) to a new T. Then A(K) takes on value A(J), whatever this value was. A(J) then gets the value located in T which was the first A(K). These commands reverse the order of two pieces of data.
110 LET J=J+1	Says for the computer to increase the value of J by one preparing the computer to continue the comparison of data.
120 IF J<=N THEN 70	If J is≤N then the computer returns to 70 to compare the new A(J) with A(K). If A(K)≤ A(J), then the program continues through steps 110,120 until either J>N or A(J)<A(K). If A(J)< A(K) steps 80-100 are repeated.
130 LET K=K+1	Sends the computer back to start the process all over again, starting with the second datum and comparing it to the rest of the list of data in the same way that it did the first item of data.
140 IF K<N THEN 60	The computer continues as long as K is less than N. When K=N then the computer goes automatically to 150.
150 LET D=A(N)-A(1)	At this point the data is an array. The computer finds the smallest datum and subtracts it from the largest.
160 FOR I=1 TO N	Statements 160-180 command the computer to punch out the array. These statements could precede statement 150.
170 PRINT A(I)	
180 NEXT I	
190 PRINT	Spaces the print-out sheet so that the information is orderly and easy to read.
200 PRINT "RANGE="D	Prints the range identifying it as such with a quotation saying this.
210 IF INT(N/2)=N/2 THEN 240	Here, the notation INT(N/2) must be explained. If these are equal then we are ready to find Median 1. If not, continue to statement 220.
220 LET M=A((N+1)/2)	Gives the computer the number of the last term of the first half of the terms of the data.
230 GO TO 250	Sends the computer to statement 250 which will print out the value of this particular A(I).


START
READ A(I)
K = 1
J = K + 1
A(K) ≤ A(J)
No
Yes
LET T = A(K)
LET A(K) = A(J)
LET A(J) = T
LET J = J + 1
J ≤ N
Yes
No
K = K + 1
K < N
Yes
No
1
1
LET D = A(N) - A(1)
A(I), D
INT(N/2) = N/2
No
Yes
M = A((N + 1)/2
M = (A(N/2) + A(N/2 + 1))/2
M
END

Flow Chart S-M-R
Sorting into an Array, Range, Median

Figure 2

students' accomplishments with that of the computer.

Lesson 3 -- The Mode

The mode is such a simple concept that five minutes of discussion is usually enough.

Definitions

1) Mode -- The mode of a set of numbers is that number which appears more frequently than any other in the set.
2) Bi-Modal -- If there are two modes in a set of data, the data is said to be bi-modal.

The exercises in the text are adequate to illustrate the idea. Exercise 3 is one that will serve as a basis for discussion: On a very easy arithmetic test given to a large number of students, would the mode of the scores likely be higher or lower than the mean? On a very difficult test, would the mode of the scores likely be higher or lower than the mean?

Included is a program which will calculate the mode as well as form an array and find the range and mean. These means are not included in these lessons, because of the nature of the class; but it may be of interest to others using a programming approach to the teaching of statistics.

```
10 READ N
20 FØR I=1 TO N
30 READ A(I)
40 NEXT I
50 FØR I=1TØ N
60 LET Q(I)=A(I)
70 NEXT I
80 GØ SUB 450
90 FØR I=1 TØ N
100 LET A(I)=Q(I)
110 NEXT I
120 LET D=A(N)-A(1)
130 FØR I=1 TØ N
140 PRINT A(I):
142 NEXT I
144 PRINT
146 PRINT "RANGE="D
150 IF INT(N/2)=N/2 THEN 180
160 LET M=A(N/2)
170 TØ TØ 190
180 LET M=(A((N+1)/2)+A((N-1)/2))/2
190 PRINT "MEDIAN="M
200 FOR I=1 TØ N
210 LET S(I)=0
220 NEXT I
230 LET I=1
240 FØR K=2 TØ N
250 IF A(I)=A(K) THEN 290
260 LET I=K
270 NEXT K
280 GØ TØ 310
290 LET S(I)=S(I)+1
300 GØ TØ 270
310 FØR I=1 TØ N
320 LET Q(I)=S(I)
330 NEXT I
340 GØ SUB 450
350 IF Q(N)=1 THEN 420
360 FØR I=1 TØ N
370 IF Q(N)=S(I) THEN 400
380 NEXT I
390 GØ TØ 10
400 PRINT A(I) "IS A MØDE"
410 GØ TØ 380
420 GØ TØ 10
430 PRINT "NØ MØDE"
440 GØ TØ 10
450 LET K=1
460 LET J=K+1
470 IF Q(K)<=Q(J) THEN 510
480 LET T=Q(K)
490 LET Q(K)=Q(J)
500 LET Q(J)=T
510 LET J=J+1
520 LET K=K+1
530 RETURN

560 END
```

Lesson 4 -- Other Means -- Geometric, Quadratic, Harmonic

This section is treated as an optional section in teaching, depending upon the background and interest of the students in a particular class. Included are programs which will calculate these three means if they are of importance to the reader.

Geometric Mean

```
10 READ N
20 FØR I=1 TØ N
30 READ X(I)
40 NEXT I
50 LET S=1
60 FØR I=1 TØ N
70 LET S=S*X(I)
80 NEXT I
90 LET G+S (1/N)
100 PRINT "GEØMETRIC MEAN="G
110 DATA 5,243,81,27,9,3
120 DATA 5,3,2,6,8,5
130 GØ TØ 10
140 END
```

Quadratic Mean

```
10 READ N
20 FØR I=1 TØ N
30 READ X(I)
40 NEXT I
50 LET S=0
60 FØR I=1 TØ N
70 LET S=S+X(I)*X(I)
80 NEXT I
90 LET Q=SQR(S/N)
100 PRINT "QUAD. MEAN="Q
110 DATA 9,6,6,7,8,8,8,11,11,12
120 DATA 8,2.5,2.5,2.5,3.5,3.5,3.8,4,4.2
130 DATA 5,3,2,6,8,5
140 GØ TO 10
150 END
```

Harmonic Mean

```
10 READ N
20 FØR I=1 TØ N
30 READ X(I)
40 NEXT I
50 LET S=0
60 FØR I=1 TØ N
70 LET S=S+1/X(I)
80 NEXT I
90 LET H=N/S
100 PRINT "HARMØNIC MEAN ="H
120 DATA 5,3,2,6,8,5
130 GØ TØ 10
140 END
```

Lesson 5 -- Mean Deviation -- Semi-Interquartile Range

Definitions

1) Mean Deviation -- a measure of variability. The mean deviation is the arithmetic mean of the absolute values of the deviations from the mean of a set of data.

   The formula for finding the mean deviation is

$$M.D. = \frac{1}{n} \sum_{i=1}^{n} \left| X_i - \overline{X} \right|$$

2) Semi-interquartile Range -- equal to one-half the range between the number, which is the last value in the first quartile and the last number in the third quartile. A quartile is one part of an array when the array is divided into four equal parts. This measure of variability is the average of the quartile deviations. The formula for finding the semi-interquartile range is

$$Q = \frac{Q_3 - Q_1}{2}$$

   The semi-interquartile range can be calculated by hand after using the computer to form an array, or it could be calculated by a program designed to follow the earlier lesson involving the array.

Since the problems in the textbook are simplified situations, they would be used to illustrate the principle involved. Hand calculating these sets of data will lead into an explanation of how the computer program works.

Computer Program for Mean Deviation

```
10 DIM X(100),D(100)
20 READ N
30 FØR I=1 TØ N
40 READ X(I)
50 NEXT I
60 LET S=0
70 FØR I=1 TØ N
80 LET S=S + X(I)
90 NEXT I
100 LET M=S/N
110 LET T=0
120 FØR I=1 TØ N
130 LET D(I)=X(I)-M
140 LET T=T + D(I)
150 NEXT I
160 LET T1 = T/N
170 PRINT"NUMBER OF VALUES"N
180 PRINT
190 PRINT
200 PRINT "S X VALUES","DEVIATIONS"
210 FØR I=1 TØ N
220 PRINT X(I),D(I)
230 NEXT I
240 PRINT
250 PRINT
260 PRINT "MEAN = "M,"MEAN DEVIATION="T1
270 DATA
300 END
```

If the problem involving automobiles is used again, the mean deviation = -1.12197×10^{-7}. Computer time to calculate is one second.

Lesson 6 -- Measure of Variability -- Standard Deviation

Definition

Standard Deviation -- The standard deviation is a measure of the average amounts by which individual items of data vary or deviate from the arithmetic mean of all the numbers that compose the data. Standard deviation is the quadratic mean of the individual deviations from the arithmetic mean. (If the class has done the section on quadratic means, then this definition is simple enough. If the class has not, then stop to define a quadratic mean at this time.) The formula for the standard deviation is:

$$\sigma = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (X_i - \overline{X})^2}$$

The problems in the text are simple, as indeed they must be. The hand calculations are still tedious. If the computer is available, then one problem from the text would be enough to understand the principle involved. Exercise 1: Compute the standard deviation of .6,.6,.7,.8,1.0, 2,1.4.

After this simple problem has been finished by the class, ask for suggestions as to how the previous computer program for mean deviation could be adapted to find standard deviation. The program for mean deviation should be shown on the overhead or written on the board with enough spaces between lines so that student suggestions could be inserted and tried to see if they will work. Hopefully, the class would discover a step or steps that would have to be inserted to handle the squaring process and finding the square root.

Program for finding the standard deviation that has been developed is as follows (this is the one we did in class):

```
10 DIM X(I),D(I)
20 READ N
30 FØR I = 1 TØ N
40 READ X(I)
50 NEXT I
60 LET S = 0
70 FØR I = 1 TØ N
80 LET S = S + X(I)
90 NEXT I
100LET M = S/N
110 LET T = 0
120 FØR I = 1 TØ N
130 LET D(I) = X(I)-M
140 LET T=T + D(I)*D(I)
150 NEXT I
160 LET T1 = T/N
170 LET T2 = SQR(T1)
180 PRINT N
190 FOR I = 1 TO N
200 PRINT X(I),D(I)
210 NEXT I
220 PRINT M,T2
230 DATA
250 END
```

Again, the example using the number of cars in the United States or some similar problem will illustrate the capabilities of the computer.

The use of standard deviation in the interpretation of data, the normal probability curve, percentages of the population falling within a certain number of standard deviations, etc., is discussed later in the chapter. This particular unit of work takes up frequency distributions, class intervals, computation of these previously discussed measures from the frequency distribution, binomial expansion and normal distribution, sampling techniques, standard error. All of this is handled in a most general way.

In summary, these lessons have been designed in a very limited manner to show the capacities of a computer. At all times during these lessons one should stress that there is nothing magic about what happens in the machine; it can only do what a human programmer tells it to do. Since these students will probably never be programmers, details of how the program works have been explained; but no requirements as to knowledge of programming should be required.

However, the occasional student could become interested in learning programming, and the instructor should be prepared to help this student progress in understanding and technique as far as possible.

USING THE HARDY-WEINBERG LAW ON A COMPUTER

by Robert L. Brown

The purpose of this paper is to show how the use of a computer could make the Hardy-Weinberg Principle of Biology more meaningful to a high school biology student. This could be accomplished in two ways: 1) by making what can be very difficult and tedious numerical computations easier (especially with a large batch of data) and 2) by showing some insight into the mathematics of the Hardy-Weinberg law itself. To understand the terms used, this material should not be introduced to students until the subject of genetics has been studied.

The Hardy-Weinberg law was first proposed independently in 1908 by the English mathematician, G. H. Hardy, and German physician, W. Weinberg. The law states that, given two alleles (A and a) which occur in the frequencies of p and q, after one generation of random mating the proportions of the three possible genotypes (AA, Aa and aa) will be in equilibrium and will stay in equilibrium generation to generation thereafter. The Hardy-Weinberg law explains one of the most striking and fundamental facts about nature: that, while bi-parental populations always contain a variety of genotypes, the population as a whole, including its variations, may continue for generation after generation without significant change. Variation, which makes evolutionary change possible, is maintained even when evolutionary change is not occurring.

For the population to stay in equilibrium, one must make some assumptions. They are: 1) the population is large, 2) that mating is random, 3) that the alleles involved have no selection value and 4) that mutations do not occur, and if they do the mutation rate is in equilibrium. Not only does the Hardy-Weinberg law enable one to study genetic traits that have no selection advantage in a population, but it enables the biologist to study alleles for traits that do have selection advantages and to compute the rate at which the frequencies of the alleles change.

To get the law in numerical form we let p = the frequency of the "A" allele (the dominant allele if there is one) and q = the frequency of the "a" allele (the recessive allele). The choice of gametes produced by a sexually reproducing organism can be represented as $p + q = 1$. The probability of the genotypes produced by two parents can be represented by the product of their gametes or $(p + q)^2 = 1^2$, which yields the Hardy-Weinberg law: $p^2 + 2pq + q^2 = 1$ where p^2 represents the AA genotypes, $2pq$ represents the Aa genotypes and q^2 represents the aa genotypes in the next generation. When there is dominance involved the AA and Aa genotypes will have the same phenotype (will look the same), but the aa homozygous recessives can be picked out of the population. This means that by finding the aa's in the population you have found the q^2's, and by taking the square root of q^2 you have found q. Since $p + q = 1$, then $p = 1 - q$; and you now have the frequencies of p and q(A and a alleles) with which you can now compute the ratios of $p^2:2pq:q^2$ (AA:Aa:aa) in the population.

As an example, take the trait found in humans called mid-digital hair (hair on the middle digit of the fingers). Genetic studies have indicated that the allele for mid-digital hair (M) is dominant to the allele for no mid-digital hair (m). A person with the trait would have the genotype MM or Mm, and a person without the trait would be mm. We will assume that mid-digital hair has no selection value and that matings involving this trait are random.

Suppose in a sample of a given population we find that 36% of the people do not have mid-digital hair. We then know that $mm = q^2 = .36$ and that $q = .6$ and then $p = .4$. It follows that $MM = p^2 = (.4)^2 = .16$ = 16% of the population, and $Mm = 2pq = 2x.4x.6 = .48$ = 48%; and we already know that $mm = q^2$ = 36% (.16 + .48 + .36 = 1 or 16% + 48% + 36% = 100%).

From the above we can compute the odds for the offspring from different types of marriages. For example, what would be the chances of two parents, both with mid-digital hair, producing a child without mid-digital hair? For this to happen, both parents would have to be of the genotype Mm or heterozygous for this trait; and the chances of their both being this way would be the answer. For the father, the chances are Mm/(MM + Mm) = 48/(16 + 48) = 3/4. The chances for the mother are the same. The chances for their both being Mm is 3/4 x 3/4 or 9/16. The odds can also be computed for other types of marriages. See Chart 1 at the end of this paper for a computer program which, when given q^2 will then compute p, q, p^2, 2pq and q^2. The program is set up such that one q^2 or many q^2 values can be fed in.

This can further be expanded to predict the frequency of marriages or matings in the population involving a given pair of alleles. Using mid-digital hair as our example, this would involve all the possible marriages of people with the genotypes MM, Mm and mm. Using $p = .4$ and $q = .6$, the frequencies of all the possible marriages would be as follows (in using this a geneticist would probably be interested in a certain type of mating rather than all of the possible matings):

MM x MM marriages = $p^2 \times p^2 = p^4 = (.4)^4$ = .0256
(since genotypes are the same, no need to specify sex)

$\left.\begin{array}{l}\text{MM x Mm}\\ \text{Mm x MM}\end{array}\right\}$ marriages = $(p^2 \times 2pq)2 = 4p^3q = 4(.4)^3 \times .6$ = .1536
(since genotypes are different, we must allow for parents being of different sex; hence, $(p^2 \times 2pq) \times \underline{2}$ above)

Mm x Mm marriages = $2pq \times 2pq = 4p^2q^2 = 4(.4)^2(.6)^2$ = .2304

$\left.\begin{array}{l}\text{MM x mm}\\ \text{mm x MM}\end{array}\right\}$ marriages = $(p^2 \times q^2)2 = 2p^2q^2 = 2(.4)^2(.6)^2$ = .1152

$\left.\begin{array}{l}\text{Mm x mm}\\ \text{mm x Mm}\end{array}\right\}$ marriages = $(2pq \times q^2)2 = 4pq^3 = 4(.4)(.6)^3$ = .3456

mm x mm marriages = $q^4 = (.6)^4$ = .1296

total 1.0000

When all the decimal frequencies total one, you know that you have not made a mistake nor left out one of the possible matings. From the above we can see that the frequency of marriages between mates with mid-digital hair and those without mid-digital hair is .1152 plus .3456 or 46% of the marriages (the frequencies for p and q used in this example were picked for convenience and do not represent the national frequencies for this pair of alleles).

In reality the frequencies are not perfect squares (q^2), and computations involving the Hardy-Weinberg law are easily adapted to computers. Charts two and three at the end of this paper show two computer programs that will determine the frequency of various possible matings when given the percentage of aa or homozygous recessives in the population (q^2).

For traits that are sex linked, like red-green color blindness and hemophilia, we must modify our approach in finding the frequencies of p and q. A color blind woman (X^aX^a) would be represented by q^2, but a color blind man (X^aY) has only one allele for this trait and would be represented by q. Since 0.6% of the females in the United States are color blind ($q^2 = 0.006$), then we would expect 8% of the males (square root of 0.006 = 0.08) to be color blind. Population samples have shown this to be true, and understanding the Hardy-Weinberg law explains why the percentage of color blind or hemophiliac males is so much greater than the same traits in females -- hence, the term sex linked traits. From the above data we can compute the odds that any woman picked at random is a carrier for the color blind allele (X^AX^a). Since $q = 0.08$, then $p = 0.92$; and the carrier women would be the 2pq's or $2 \times .92 \times .08 = .1472$ or about 15%. The odds are 15/100 that any woman picked at random is a carrier for color blindness.

The Hardy-Weinberg law is also very useful in working with populations with alleles where one allele does have selection value over the other. In this case, the population would not be in equilibrium for this trait; in fact, it would be evidence that natural selection is favoring certain genotypes.

Next we will look at selection against dominant and recessive genotypes. To simplify matters, we will assume that the alleles are fully penetrant and that the genotypes are lethal or such that the organism never reproduces (dies before reproductive maturity, or if it reaches maturity is unable to reproduce and thus unable to pass the "bad" allele on to the next generation).

In the case of the bad dominant allele that appears through mutation (remember if either of the parents would have shown the trait, they would not have reproduced), the allele would be eliminated in one generation; since the organism would never reproduce and thus never pass the bad allele on to the next generation. The genotypes are almost always heterozygous or Aa for the trait, since the A allele would arise by mutation. Thus, in one generation the bad trait is eliminated except for new mutations from the recessive to the dominant allele. In reality for this case the frequency of the bad allele will be nearly equal to the mutation rate of $a \rightarrow A$. In any case, the allele is quickly eliminated from the population gene pool.

Selection against a bad recessive allele is much less effective. The lower the frequency of the bad allele, the less the change in frequency of the genotype from generation to generation. Remember, we are assuming that these aa genotypes do not reproduce. Most of the aa bad genotypes are produced by Aa heterozygous parents who are normal for the trait. The exceptions would have to involve mutations. When we assume that certain types of mental deficiencies are inherited, we find that most of the mentally defective children come from normal parents -- not from mentally defective parents. The sterilization of all mentally defective people at birth would not lead to a significant reduction in mentally defective offspring the next generation.

If with full penetrance the frequency of aa is q_o^2, and if no affected person reproduces, the number of aa individuals in the next generation can be calculated from the frequency q_1 of the a alleles in the reproducing population. The total frequency of the reproducing population is the sums of AA and Aa = $p_o^2 + 2p_oq_o$, while the a alleles in this population is 1/2 of the frequency of the Aa genotypes, or P_oq_o.

The ratio of a among all alleles after disregarding the non-reproducing genotypes is:

$$q_1 = \frac{p_oq_o}{p^2+2p_oq_o} = \frac{q_o}{p_o+2q_o} \text{ where } q_o = \text{initial frequency,}$$

q_1 = 1st generation, q_2 = 2nd generation, etc.

Since $p_o = 1 - q_o$ this becomes:

$$q_1 = \frac{q_o}{1 + q_o} \quad \text{(substitute } 1\text{-}q_o \text{ for } p_o \text{ above)}$$

and the frequency of aa genotypes in the next generation is:

$$(q_1)^2 = \left(\frac{q_o}{1 + q_o}\right)^2$$

If we start with an initial frequency of 1% (.01) for the aa (a_o^2),

$$\text{then } q_1^2 = \left(\frac{\sqrt{.01}}{1 + \sqrt{.01}}\right)^2 = 83\% \text{ and } q_2^2 = \left(\frac{\sqrt{.0083}}{1 + \sqrt{.0083}}\right)^2 = .69\%$$

Thus, in our first generation we only have a reduction of .17%; and moving to the second generation we get a reduction of only .14%. The reduction becomes less and less as the frequency of q becomes less. See Graph 1 at the end of the paper.

In general terms, after two generations:

$$\text{(D1)} \quad q_1 = \frac{q_o}{1 + q_o} \qquad\qquad \text{(D2)} \quad q_2 = \frac{q_1}{1 + q_1}$$

by substituting the right side of (D1) for q_1 in (D2) above, we get:

$$q_2 = \frac{\dfrac{q_o}{1 + q_o}}{1 + \dfrac{q_o}{1 + q_o}} = \frac{q_o}{1 + 2q_o}$$

or a general formula to find the reduction for several generations:

$$(q_n)^2 = \left(\frac{q_o}{1 + nq_o}\right)^2$$

where $(q_n)^2$ is equal to the frequency of aa genotypes expected from the nth generation. See Chart 4 for a computer program that carries this out for 10 generations. The graph in Graph 1 was plotted from this program starting with an initial frequency for q_o^2 of 1%. After 10 generations, the frequency of aa genotypes has been reduced from 1% to only .25%. Graphs 2 and 3 show the result of 10 generations starting with initial frequencies of .1%

and .01% respectively for aa genotypes.

This result shows that the reduction in frequency of aa genotypes becomes less and less in each successive generation and that after 10 consecutive generations of complete selection, it is still 1/4 of the initial frequency of 1%. To reduce it to 1/10 (.1%) would require 22 generations, from .1% to .01% would require 68 more generations, from .01% to .001% would require 216 more generations, and from .001% to .0001% would require 684 more generations. When one attempts to compute the above percentages one soon realizes the advantages of computers, which will carry out the numerical operations for whatever generation desired in less than a second.

This has been an attempt to show how computers could aid in the teaching of high school biology. There are many ideas in biology which could be aided by computer techniques. Some areas involved today are information storage, growth forms and patterns, growth rates, taxonomic problems and many types of biochemical and physiological problems, to mention only a few.

A good reference for applications of the Hardy-Weinberg law to human populations is: STERN, Curt, 1960. Principles of Human Genetics, 733 pp., W. H. Freeman and Co., San Francisco.

Chart 1

Program

```
10 PRINT "Q", "P", "P2", "2PQ", "Q2"
20 READ Q2
30 LET Q = SQR(Q2)
40 LET P = 1 - Q
50 LET P2 = P*P
60 LET P3 = 2*P*Q
70 PRINT Q, P, P2, P3, Q2
80 GØ TØ 20
90 DATA .00005, .04, .09, .16, .25, .36, .49, .64, .81
100 END
```

Answers

```
HARDY      13:41     PX FRI 06/30/67
                              AA              Aa                  aa
Q               P             P2              2PQ                 Q2
                                              .0140421
 7.07107  E-3    .992929       .985908        1.40421 E-2          .00005
 .2              .8            .64            .32                 .04
 .3              .7            .49            .42                 .09
 .4              .6            .36            .48                 .16
 .5              .5            .25            .5                  .25
 .6              .4            .16            .48                 .36
 .7              .3            .09            .42                 .49
 .8              .2            .04            .32                 .64
 .9              .1            .01            .18                 .81

ØUT ØF DATA IN 20
```

A computer program written in "BASIC" computer language to find the values of q, p, p^2, and 2pq when given q^2 using the Hardy-Weinberg law. To enter new values for q^2 you would change the data in line 90, the DATA statement. Note that the value of q has to reach 0.7 for 50% of the population to show the recessive trait. For more information on setting up a program in "BASIC" language, see the 1967 edition of the General Electric "BASIC" Language reference manual put out for their computer time-sharing service.

Chart 2

Program

```
10 READ Q2
20 LET Q = SQR(Q2)
30 LET P = 1 - Q
40 LET P1 = P*P*P*P
50 LET P2 = 4*P*P*P*Q
60 LET P3 = 4*P*P*Q*Q
70 LET P4 = 2*P*P*Q*Q
80 LET P5 = 4*P*Q*Q*Q
90 LET P6= Q*Q*Q*Q
100 PRINT Q2,Q, P, P1, P2, P3, P4, P5, P6
110 DATA .3, .04
120 GØ TØ 10
130 END
```

Answers

		Population 30% aa	Population 4% aa
(Q2) q^2	=	0.3 (30%)	0.04 (4%)
(Q) q	=	0.547723	0.2
(P) p	=	0.452277	0.8
(P1) AAxAA marriages	=	0.0418427	0.4096
(P2) AAxAa marriages	=	0.202691	0.4096
(P3) AaxAa marriages	=	0.245466	0.1024
(P4) AAxaa marriages	=	0.122733	0.0512
(P5) Aaxaa marriages	=	0.297267	0.0256
(P6) aaxaa marriages	=	0.0900	0.0016

A computer program to find the frequencies of all the possible matings involving alleles which have no selection value. The ( ) represent the symbols for the above statements in the computer program. For other values of q^2 enter new data for line 110, the DATA statement (convert all percentages into their decimal equivalents). The program is set up in such a way that one or a hundred bits of data could be entered at one time.

Chart 3

Program

```
10 READ Q2
20 LET QP SQR(Q2)
30 LET P = 1 - Q
40 LET P1 = (P*P*P*P) + (4*P*P*P*Q) + (4*P*P*Q*Q)
50 LET P2 = (2*P*P*Q*Q) + (4*P*Q*Q*Q)
60 LET P3 - Q*Q*Q*Q
70 PRINT Q2, Q, P, P1, P2, P3
80 DATA .23, .04, .09, .16, 125, .36, .49, .64, .81
90 GØ TØ 10
100 END
```

Answers

```
q^2=.23        q=.479583       p=.520417       P1=.5929        P2=.3542
P3=.0529

   .04           .2                .8              .9216          .0768
   .0016

   .09           .3                .7              .8281          .1638
   .0081

   .16           .4                .6              .7056          .2688
   .0256

   .25           .5                .5              .5625          .3750
   .0625

   .36           .6                .4              .4096          .4608
   .1296

   .49           .7                .3              .2601          .4998
   .2401

   .64           .8                .2              .1296          .4608
   .4096

   .81           .9                .1              .0361          .3078
   .6561

ØUT ØF DATA IN 10
```

A computer program to find the frequencies of all the possible matings of three types: 1) both parents show the dominant trait, 2) one parent shows the dominant trait, and one parent the recessive trait; and 3) both parents show the recessive trait. P1 in the program is that part of the expected matings where both parents show the dominant trait; P2 is one parent dominant and one parent recessive; P3 is both parents recessive. The answers repeat for a given q^2 in the order indicated (q^2, q, p, P1, P2, and P3).

Chart 4

Program

```
5 PRINT "GENERATIØN", "FREQUENCY ØF GENØTYPE IN PERCENT"
10 READ Q2
20 LET Q(0) = SQR(Q2)
30 FOR N = 1 TØ 10
40 LET Q(N) = (Q(0)/(1 + N*Q(0)))↑2*100
50 PRINT N, Q(N)
60 NEXT N
70 DATA .01, .001, .0001
80 GØ TØ 10
90 END
```

Answers

```
GENERATIØN       FREQUENCY ØF GENØTYPE IN PERCENT
 1                .826446
 2                .694444
 3                .591716
 4                .510204
 5                .444444
 6                .390625
 7                .346021
 8                .308642
 9                .277008
 10               .25
 1                9.39633 E-2
 2                8.84571 E-2
 3                8.34212 E-2
 4                7.88033 E-2
 5                7.45586 E-2
 6                7.06477 E-2
 7                6.70368 E-2
 8                6.36957 E-2
 9                6.05983 E-2
 10               5.77215 E-2
 1                9.80296 E-3
 2                9.61169 E-3
 3                9.42596 E-3
 4                9.24556 E-3
 5                9.07029 E-3
 6                8.89996 E-3
 7                8.73439 E-3
 8                8.57339 E-3
 9                8.41680 E-3
 10               8.26446 E-3

ØUT ØF DATA IN  10
```

A computer program to calculate complete selection against a recessive genotype. These answers were used to plot Graphs 1, 2, and 3. Each q^2 listed in the data (% of aa genotypes changed to decimal equivalents) was run for 10 generations. To increase or decrease the number of generations, you would change the 10 in line 30 to the number of generations desired. The E-2 found in the answers means the numerical value X 10^{-2}, as the computer puts the answers in this form to give the benefit of more significant digits. In line 40 the equation

$$(q_n)^2 = \left(\frac{q_o}{1 + nq_o}\right)^2$$

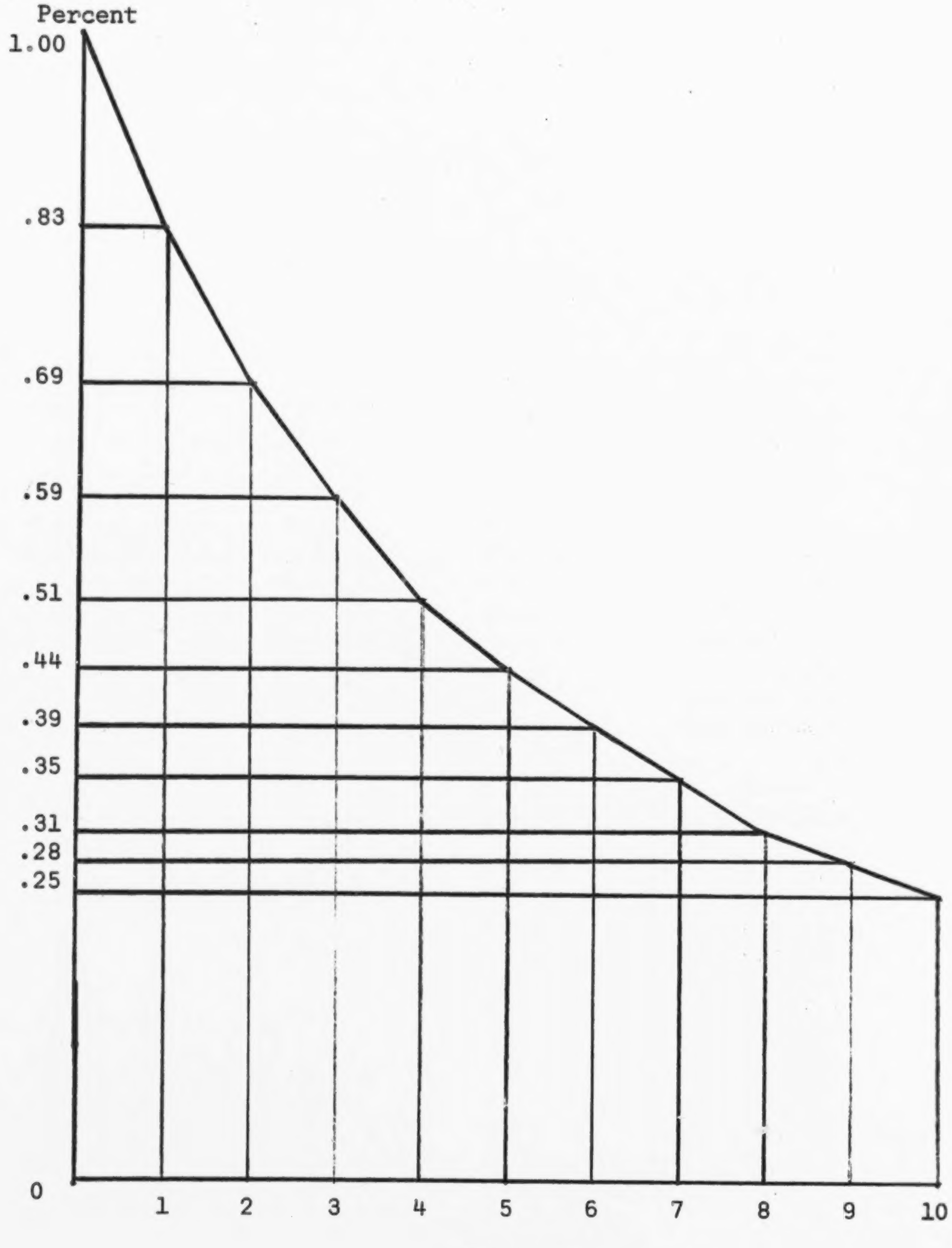
was multiplied by 100 to put the answers in % form to make graphing easier. The computer used one second of time to compute the 30 answers given. These were stored in a memory unit and then read out by the teletype after all values had been computed. To get these answers by hand would require several hours.

Chart 5

```
90 DATA .0001,.0002,.0003,.0004,.0005,.0006,.0007,.0008,.0009
91 DATA .001,.002,.003,.004,.005,.006,.007,.008,.009
92 DATA .01,.02,.03,.04,.05,.06,.07,.08,.09
93 DATA .1,.2,.3,.4,.5,.6,.7,.8,.9,1
RUN

HARDY      14:41     PX MØN 07/03/67

Q                 P            P2                2PQ                Q2
 .01               .99          .9801             .0198              .0001
 1.41421 E-2       .985858      .971916           2.78843 E-2        .0002
 1.73205 E-2       .982679      .965659           .034041            .0003
 .02               .98          .9604             .0392              .0004
 2.23607 E-2       .977639      .955779           4.37214 E-2        .0005
 2.44949 E-2       .975505      .95161            4.77898 E-2        .0006
 2.64575 E-2       .973542      .947785           .051515            .0007
 2.82843 E-2       .971716      .944231           5.49685 E-2        .0008
 .03               .97          .9409             .0582              .0009
 3.16228 E-2       .968377      .937754           6.12456 E-2        .001
 4.47214 E-2       .955279      .912557           8.54427 E-2        .002
 5.47723 E-2       .945228      .893455           .103545            .003
 6.32456 E-2       .936754      .877509           .118491            .004
 7.07107 E-2       .929289      .863579           .131421            .005
 7.74597 E-2       .92254       .851081           .142919            .006
 .083666           .916334      .839668           .153332            .007
 8.94427 E-2       .910557      .829115           .162885            .008
 9.48683 E-2       .905132      .819263           .171737            .009
 .1                .9           .81               .18                .01
 .141421           .858579      .737157           .242843            .02
 .173205           .826795      .68359            .28641             .03
 .2                .8           .64               .32                .04
 .223607           .776393      .602786           .347214            .05
 .244949           .755051      .570102           .369898            .06
 .264575           .735425      .54085            .38915             .07
 .282843           .717157      .514315           .405685            .08
 .3                .7           .49               .42                .09
 .316228           .683772      .467544           .432456            .1
 .447214           .552786      .305573           .494427            .2
 .547723           .452277      .204555           .495445            .3
 .632456           .367544      .135089           .464911            .4
 .707107           .292893      8.57864 E-2       .414214            .5
 .774597           .225403      5.08067 E-2       .349193            .6
 .83666            .16334       2.66799 E-2       .27332             .7
 .894427           .105573      1.11456 E-2       .188854            .8
 .948683           5.13167 E-2  2.63340 E-3       9.73666 E-2        .9
 1                 0            0                 0                  1

ØUT ØF DATA IN  20

TIME:   3 SECS.
```

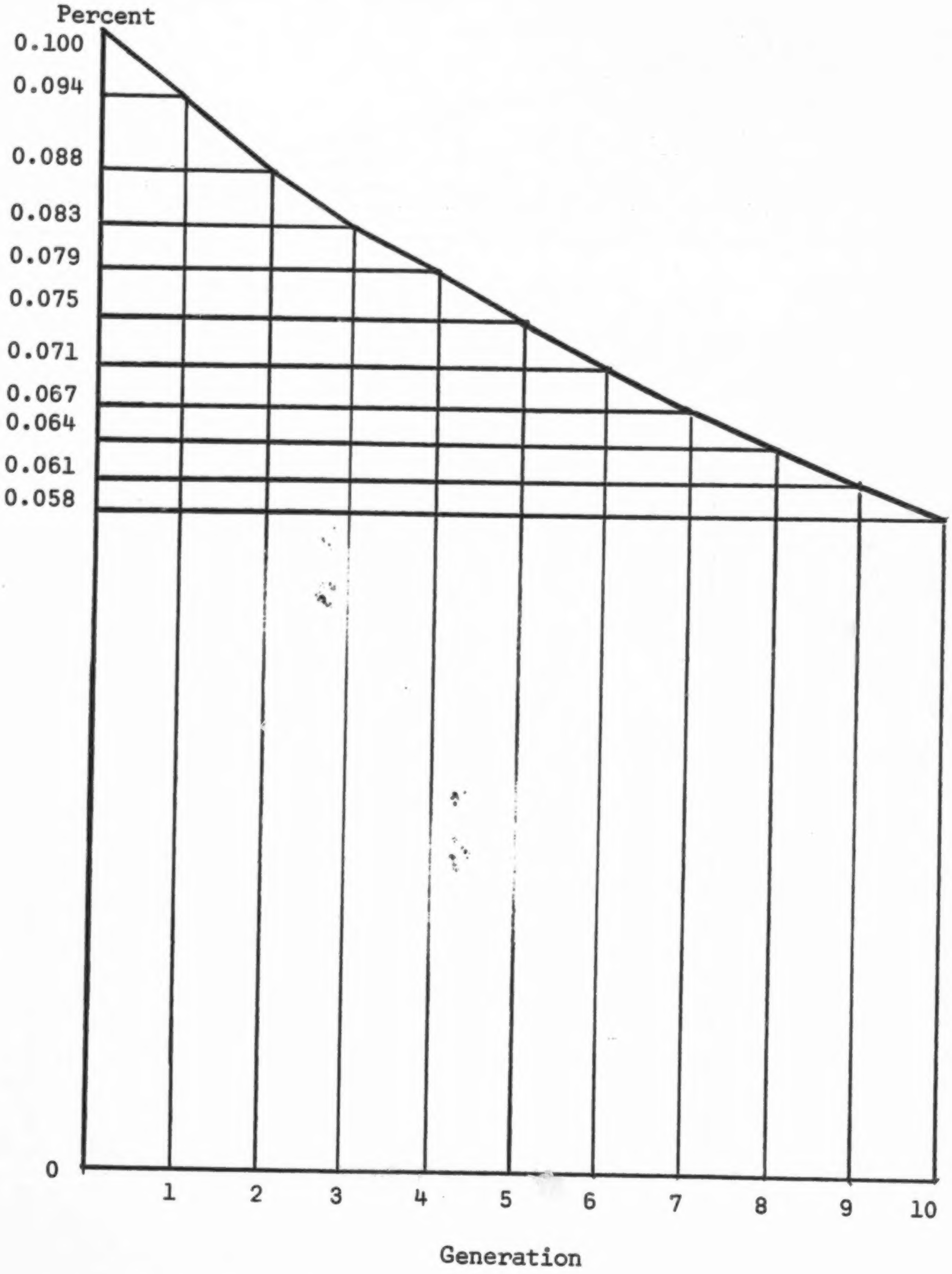
This is the same program as listed in Chart 1 with more data. Only the new data and answers are listed. This chart contains or approximates many of the q^2 values used in Hardy-Weinberg problems. Remember that the q^2 percentages from the population must be converted into their respective decimal equivalents.

Graph 1



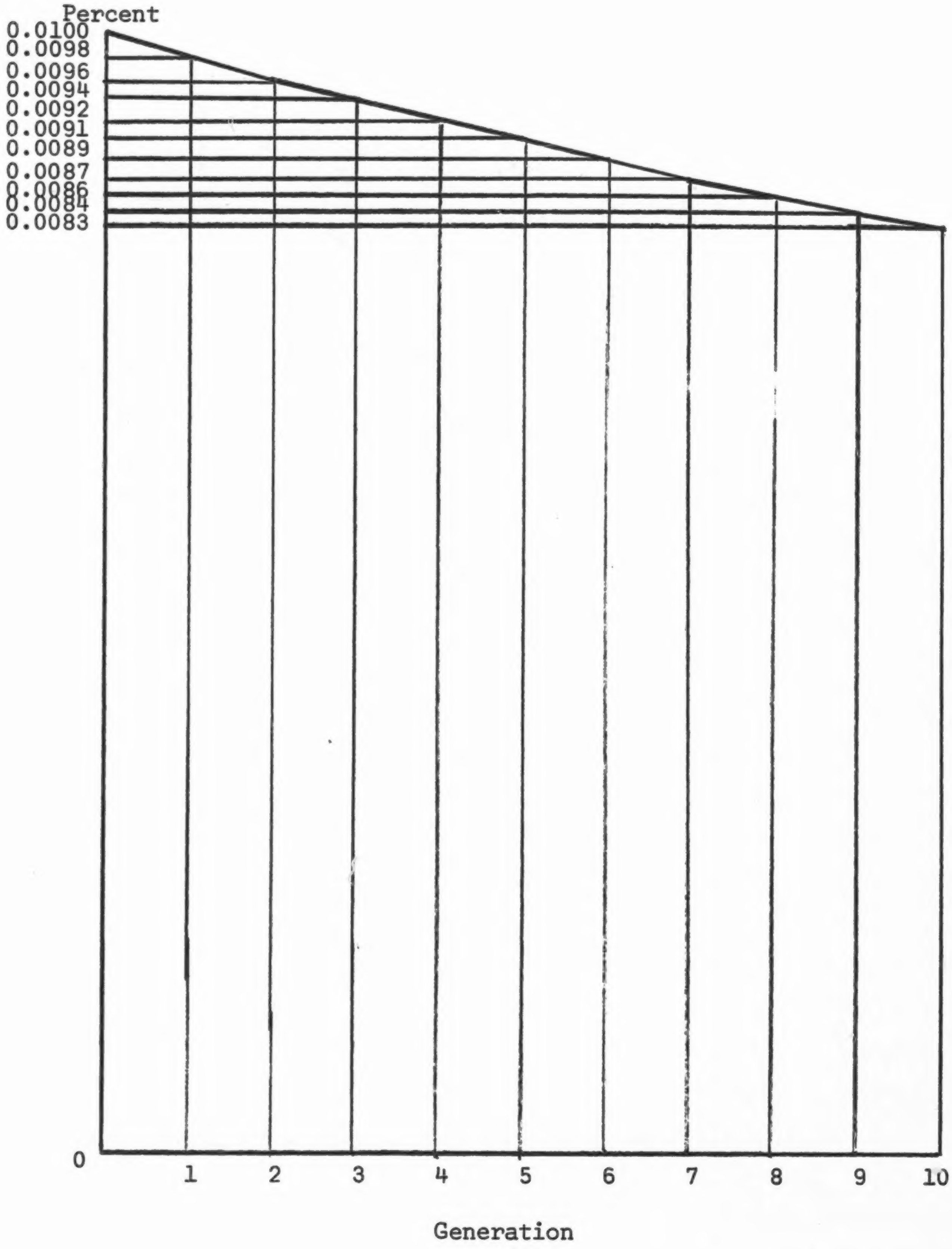
Complete selection against a recessive genotype. Initial frequency of genotype in the population was 1%. After 10 generations, the frequency has been reduced from 1% to .25%. After 10 generations, the frequency is still 1/4 the initial frequency.

Graph 2



Complete selection against a recessive genotype. Initial frequency of the genotype in the population was 0.1%. After 10 generations, the frequency has been reduced from 0.1% to 0.058%. After 10 generations, the frequency is still over 1/2 the initial frequency.

Graph 3



Complete selection against a recessive genotype. Initial frequency of the genotype in the population was 0.01%. After 10 generations, the frequency has been reduced from 0.01% to 0.0083%. After 10 generations, the frequency is still over 4/5 the initial frequency. (The lines do not necessarily become closer as the % drops due to rounding off. See Chart 4 for the actual data before rounding off.)

PYTHAGOREAN TRIADS

by Dewayne N. Burgdorff and Jay E. Niebur

Introduction

The main purpose of this lesson is to show the relationship that exists between the numbers which satisfy the Pythagorean Theorem. These numbers are called Pythagorean triples or triads. All triads or triples are integers such that the sum of the squares of two will always equal the third. In other words, if the integers are a, b, and c, then by applying the Pythagorean Theorem $a^2 + b^2$ should equal c^2.

Historical Aspects of Theorem

The Pythagorean Theorem is one of the most important theorems in the realm of geometry. Not only does it apply to plane or solid geometry, but it has application in analytic geometry, as well as in the field of trigonometry. The origin of this famous theorem dates back before Pythagoras generalized it.

The Egyptians used the relationship of the 3, 4, 5 right triangle to lay out their land after it was flooded by the Nile River each year. They used ropes measured off with these dimensions and from this acquired the names of "rope stretchers." Also, about the same time the Egyptians were using this relationship, the Hindus in India had developed more advanced ideas concerning right triangles. Besides using the 3,4,5, their right triangles included such measurable values as the 12,16,20; 15,20,25; 5,12,13; 15,36,39; 8,15,17; and the 12,35,37. Neither the Egyptians nor the Hindus tried to prove why this relationship held true.

It was about 540 B.C. before Pythagoras used the Egyptian rule of thumb, $3^2 + 4^2 = 5^2$, to generalize this now famous theorem. It has not been proven that Pythagoras truly was the one who developed this theorem or whether some other Greek found the answer. Be that as it may, this mathematical rule will go down in history as the Pythagorean Theorem.

Since the Pythagorean Theorem was formulated, many individuals have tried to prove this theorem. For example, in 1940 Elisha S. Loomis, a mathematics professor at Baldwin Wallace University, wrote a book titled, The Pythagorean Proposition. It contained over 370 formulized proofs.

Probably the most famous of all these proofs was a geometrical proof discovered by Euclid. It has been stated that there is no better logically derived proof than that of Euclid. Following is this famous proof and three other simple algebraic proofs.

THE PYTHAGOREAN THEOREM
(EUCLID'S PROOF)

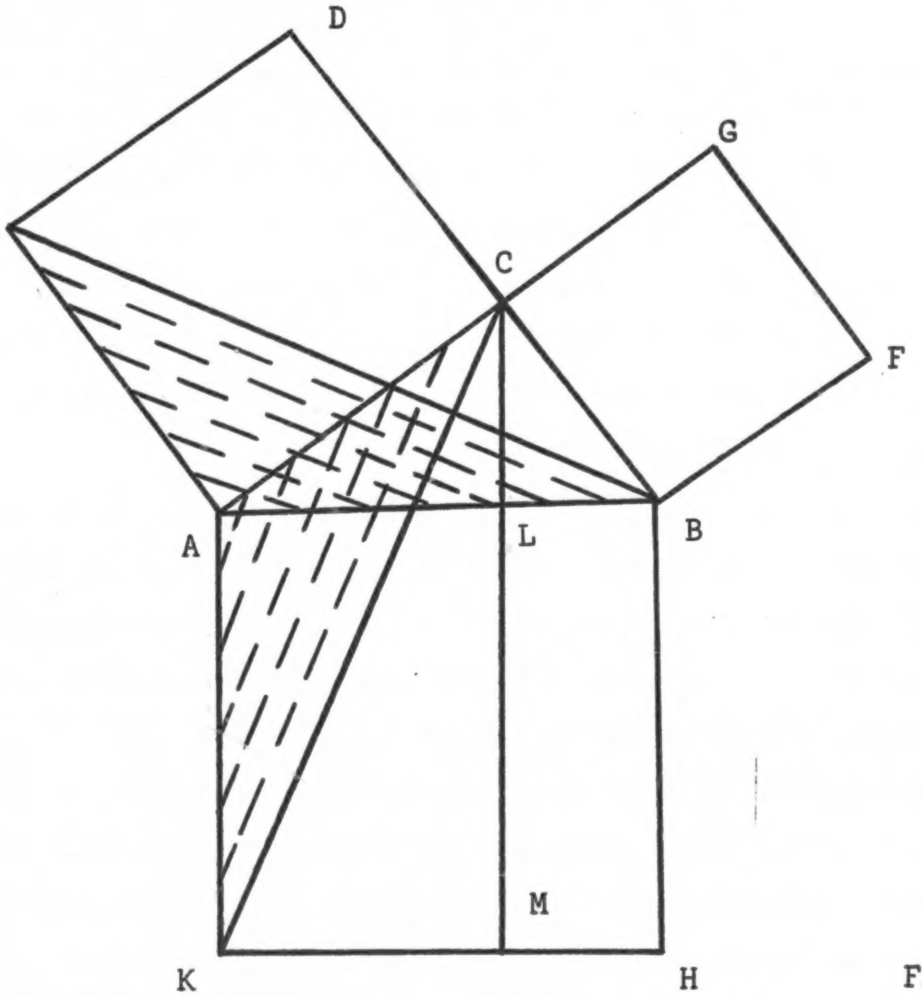


Figure 1

Given: Triangle ABC, Angle ACB is a right angle, squares constructed on Sides AB, BC, and AC.

Prove: Square ABHK = Square ACDE + Square BFGC

1.	Draw CM // AK	1.	A line can be drawn // to a given line.
2.	Draw CK and BE	2.	Two points determine a line.
3.	In triangles KAC and BAE, AK = AB; and AC = AE.	3.	Adjacent sides of a square are equal.
4.	/ KAC = / EAB	4.	If equals are added to equals their sums are equal.
5.	Triangles KAC and ABE are congruent.	5.	S.A.S. = S.A.S.
6.	DCB is a straight line.	6.	If two angles are supplementary, their exterior sides lie in a straight line.
7.	Triangle BAE and rectangle ACDE have same base and alt.	7.	Parallel lines are everywhere equidistant.
8.	BAE = $\frac{1}{2}$ rectangle ACDE	8.	If a triangle and a parallelogram have same bases and altitudes, the area of the triangle equals one-half that of the parallelogram.
9.	KAC and rectangle KMLA have same bases and altitude.	9.	Same as Number 7.
10.	KAC = $\frac{1}{2}$ rectangle KMLA	10.	Same as Numbers 5 and 8.
11.	$\frac{1}{2}$ rectangle KMLA = $\frac{1}{2}$ square ACDE.	11.	Substitution axiom.
12.	Rectangle KMLA = Square ACDE.	12.	If equals are multiplied by equals their products are equal.

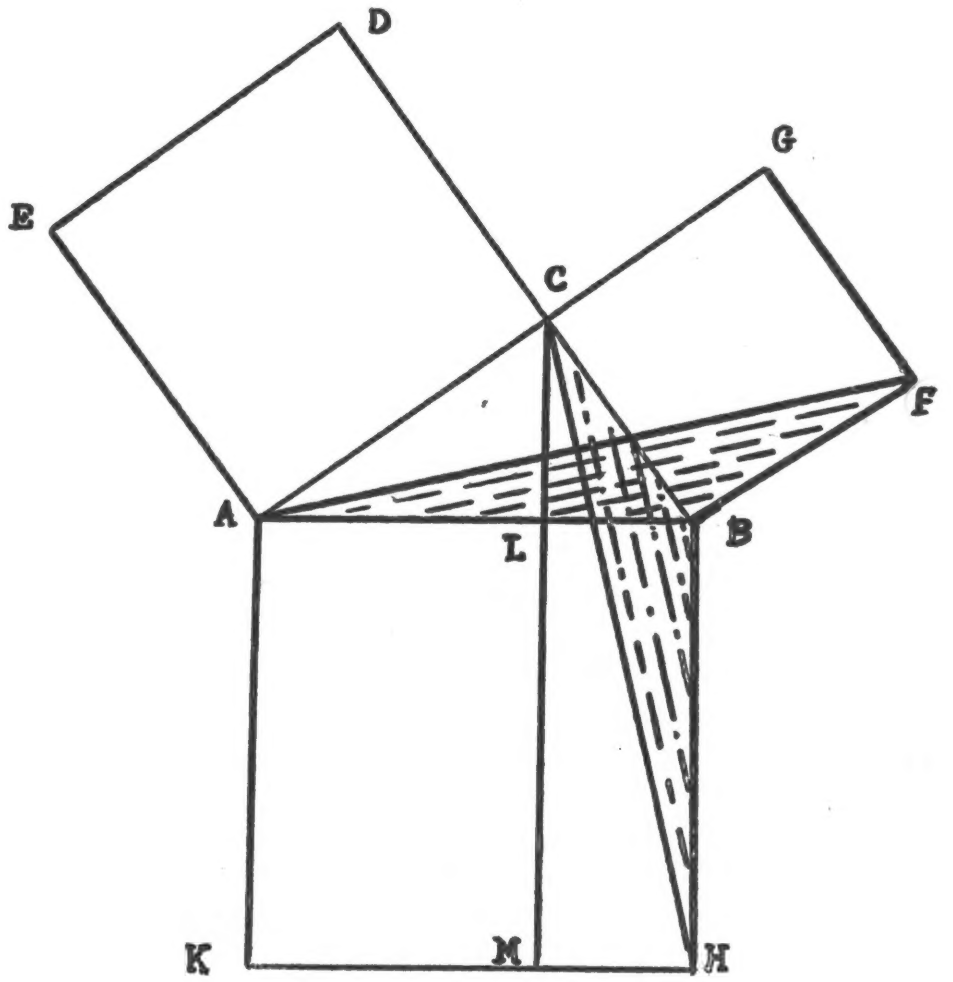


Figure 2

13.	Draw AF and CH	13.	Two points determine a line.
14.	In triangles ABF and HBC, BC = BF; and AB = BH.	14.	Adjacent sides of a square are equal.
15.	$\angle$ ABF = $\angle$ HBC.	15.	If equals are added to equals, their sums are equal.
16.	Δ ABF ≅ Δ HBC.	16.	S.A.S. = S.A.S.
17.	ACG is a straight line.	17.	If two angles are supplementary, their exterior sides lie in a straight line.
18.	Δ ABF and rect. BFGC have same bases and altitude.	18.	Parallel lines are everywhere equidistant.
19.	Δ ABF = $\frac{1}{2}$ rect. BFGC.	19.	Same as Number 8.
20.	Δ HBC and rect. MHBL have same bases and altitude.	20.	Same as Number 18.
21.	Δ HBC = $\frac{1}{2}$ rect. MHBL.	21.	Same as Numbers 5 and 8 inclusive.
22.	$\frac{1}{2}$ sq. BFGC = $\frac{1}{2}$ rect. MHBL.	22.	Substitution axiom.
23.	Sq. BFGC = rect. MHBL.	23.	Multiplication axiom.
24.	KMLA + MHBL = ACDE + BFGC.	24.	Addition axiom.
25.	KMLA + MHBL = sq. KHBA.	25.	The whole is equal to the sum of its parts.
26.	Sq. KHBA = Sq. ACDE + BFGC.	26.	Substitution axiom.

Another means by which this theorem may be expressed is:

$$a^2 + b^2 = c^2$$

Method I. Given four identical right triangles placed in such a location to form a given square ABCD (as shown in Figure 3).

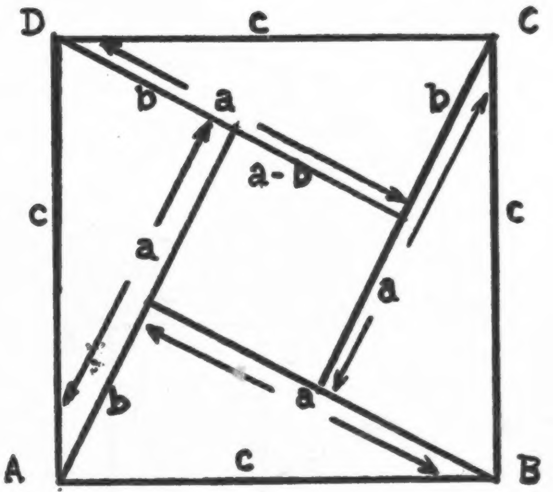


Figure 3

Proof: The square ABCD has sides c, and its area is c^2. The area of the square is made up of the four triangles plus the smaller inside square. The area of the smaller square is equal to $(a - b)^2$. The area of the four triangles is $4(\frac{1}{2} ab)$. The area of the large square, c^2, equals the sum of the smaller square, $(a - b)^2$ and the four triangles, $4(\frac{1}{2} ab)$. Algebraically:

$$c^2 = (a - b)^2 + 4(\tfrac{1}{2} ab)$$

which simplifies to

$$c^2 = (a^2 - 2ab + b^2) + 2ab$$

Finally

$$c^2 = a^2 + b^2$$

Method II. (See Figure 4.)

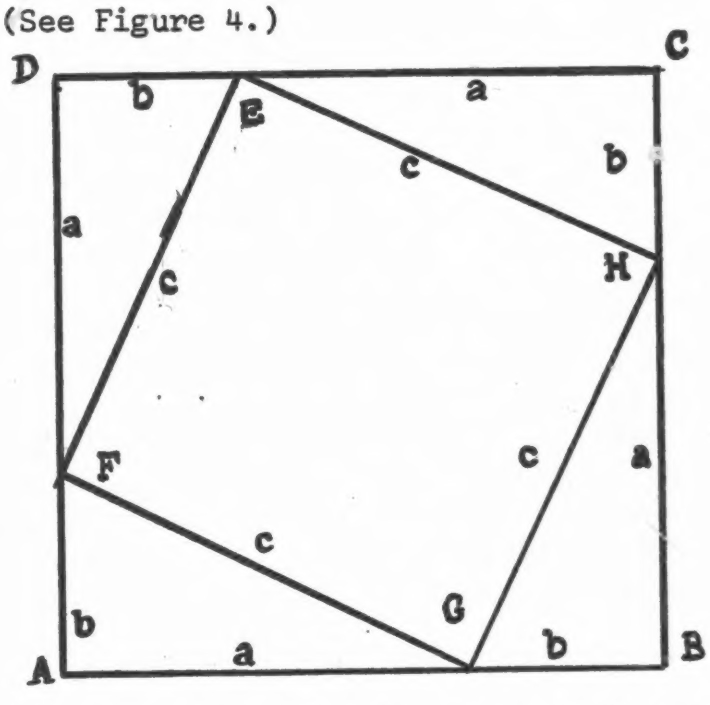


Figure 4

Proof: Use the same four triangles used in Method I. Then, on the basis of the area, you have the given relationship:

$(a + b)^2 = 4(\frac{1}{2} ab) + c^2$ which simplifies to

$a^2 + 2ab + b^2 = 2ab + c^2$ Finally

$a^2 + b^2 = c^2$

Method III. Given right triangle ABC with altitude CD drawn to the hypotenuse of the triangle (as shown in Figure 5). The two smaller triangles are similar.

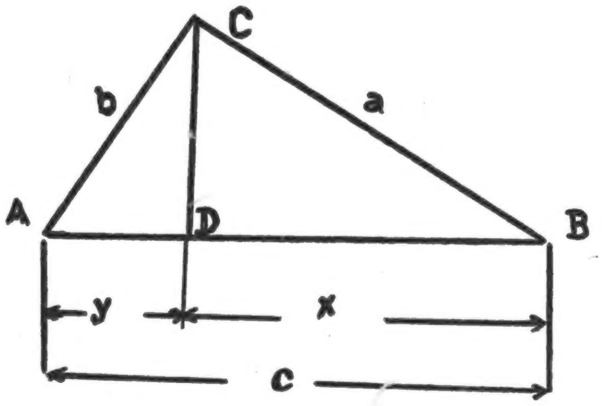


Figure 5

Proof: Since triangles ACB, ADC, and CDB are similar, their corresponding sides are proportional. From these proportionalities, the following proportions are formed:

1) $x/a = a/c$ 2) $y/b = b/c$

but it is also true that:

3) $x + y = c$.

From the equations 1) and 2), it can be found that

$x = a^2/c$ and $y = b^2/c$.

By substituting the expressions for x and y into equation 3), You obtain: $a^2/c + b^2/c = c$

Then by clearing fractions and multiplying both sides by c, you will have the Pythagorean Theorem.

$$a^2 + b^2 = c^2$$

Pythagorean Triads

Pythagorean triads are three integers which represent the lengths of three sides of a right triangle. In this discussion the letters a and b are the legs of any right triangle, and the letter c will correspond to the hypotenuse of the same right triangle.

There are two main classifications of Pythagorean triads. They are classified as multiple values or unique. The multiple valued numbers are obtained from unique triads which have been multiplied by a constant k such that $ka^2 + kb^2 = kc^2$.

In discussing triads which are unique, there exist certain relationships which hold true. They are as follows:

1. In unique triads, no two numbers may have a common factor. They are relatively prime numbers.

2. If a, b, and c are triads, then a and b cannot both be odd.
3. If a, b, and c are triads, c must always be odd; and of the other two integers, one must be even and the other odd.
4. When the sides of a right triangle are integers, the perimeter of the triangle will be an even number; and its area will also be even.

Since the development of the Pythagorean Theorem, mathematicians through the ages have tried to develop a general formula for the formation of these triads. Some of the men worthy of mention in addition to Pythagoras (582-501 B.C.), are Plato (429-348 B.C.), Euclid (300 B.C.), Maseres (1721-1824), Leonard E. Dickson, and Dr. Artemus Martin.

Following is a list of the rules formulated by the preceding men. Although there is some similarity in all of the rules, each of these men are given credit for finding formulas which will give Pythagorean triads.

The rules of finding integral values of a, b, and c are stated in order of their discovery.*

1. *Rule of Pythagoras*: Let n be any odd number, then n, $(n^2 - 1)/2$ and $(n^2 + 1)/2$ are three such numbers. For $n^2 + ((n^2 - 1)/2)^2 = ((n^2 + 1)/2)^2$.
2. *Plato's Rule*: Let m be any even number divisible by 4; then m, $m^2/4 - 1$ and $m^2/4 + 1$ are three such numbers. For $m^2 + ((m^2/4 - 1)) = ((m^2/4 + 1))^2$.
3. *Euclid's Rule*: Let x and y be any two even or odd numbers such that x and y contain no common factor greater than 2 and xy is a square. Then $\sqrt{xy}$, $(x - y)/2$ and $(x + y)/2$ are three such numbers. The rule may then be expressed as: $(\sqrt{xy})^2 + ((x - y)/2)^2 = ((x + y)/2)^2$.
4. *Rule of Maseres*: Let m and n be any two even or odd numbers, m is greater than n and $(m^2 + n^2)/2n$ is an integer. Then m^2, $(m^2 - n^2)/2n$ and $(m^2 + n^2)/2n$ are three such numbers. For $m^2 + (m^2 - n^2)/2n = ((m^2 + n^2)/2n)^2$.
5. *Dickson's Rule*: Let m and n be any two prime integers, one even and the other odd, m greater than n and 2mn a square. Then $m + \sqrt{2mn}$, $n + \sqrt{2mn}$, and $m + n + \sqrt{2mn}$ are three such numbers. For $(m + \sqrt{2mn})^2 + (n + \sqrt{2mn})^2 = (m + n + \sqrt{2mn})^2$.

---

*Elisha S. Loomis, *The Pythagorean Proposition*, (Michigan; Edward Brothers, Inc., 1940), pp. 19-20.

6. Martin's Rule: Let $(x + y)^2 = (x - y)^2 + 4xy$. Assume $4xy$ is a square. If you assume $x = mp^2$ and $y = mq^2$ and we have $4xy = 4m^2p^2q^2$, which is a square number for all values of m, p, and q, then by substitution we arrive at $(p^2 - q^2)^2 + (2pq)^2 = (p^2 + q^2)^2$.

A flow chart is a diagram of the steps followed by a computer in the solution of a problem.

In the flow chart for the problem called TRIADS (see Figure 6) the odd integer 3 is placed in the computer as N and is used to calculate A, B, and C from the formulas $A = N^2$, $B = (\frac{N^2 - 1}{2})^2$, and $C = (\frac{N^2 + 1}{2})^2$. If C does not equal A + B, then the values calculated for N = 3 would be discarded; and the computer would take the next odd integer (N = 5) and perform the same calculations. If C does equal A + B, as it does when N = 3 (C = 25, B = 16, A = 9 so 25 = 16 + 9), then the computer will output the triad $\sqrt{A}$, $\sqrt{B}$, and $\sqrt{C}$; or in this case 3, 4, and 5, which is a right triangle. This is a unique triad, and all multiples of the 3, 4, 5 combination will be right triangles also. Therefore, beginning with a multiplier (I) of 2, we take I x $\sqrt{A}$, I x $\sqrt{B}$ and I x $\sqrt{C}$, or 2 x 3, 2x 4, and 2 x 5, which is another right triangle with sides 6, 8, and 10.

Letting I = (previous)I x 1, a new I is generated so that I now will equal 3. This multiplier 3 is taken times the unique triad, producing a 9, 12, 15 right triangle. I will be incremented again and more multiples found until A1 becomes greater than 100. In that case, the computer goes to the next step and decides if N < = 100. N is still 3, which is an odd number less than 100; so the program goes to connector A and selects a new N. This new N will be 5, and the computer will perform all the previous steps but using N = 5. After N = 99 has been calculated, the program will follow connector C and will receive an even number. N = 4 will be the first situation, and values for A, B, and C will be calculated using formulas $A = N^2$, $B = (\frac{N^2 - 4}{2})^2$, and $C = (\frac{N^2 + 4}{2})^2$.

The program will proceed as before until N > 100.

In the flow chart called GUESS (see Figure 7), an integer A, then an integer B will be placed in the computer. Calculating $X = A^2$ and $C = \sqrt{a^2 + B^2}$, the computer decides if $C^2 - (C - 1)^2 > X$. If the answer is yes, a new value for A will be taken and new calculations begun. If the answer is no, the computer will decide if C is less than or greater than the integer C. If the answer is yes, meaning C is not an integer, then a new value for B will be selected, and different calculations made using the new B. If the answer is no, meaning C is an integer, then the values for A, B, and C will be printed; and these three numbers will be triads.

Through the ages, many hours have been spent in rigorous methods of solutions for the calculations of triads. Now, by means of a computer, one is able to obtain, in a relatively short amount of time, many combinations of triads in relation to the Pythagorean Theorem. The data was obtained by using the GE 265 computer. By constructing a flow chart, the following problems were programmed.

Problem 1 -- By applying the rules of Pythagoras and Plato, try to obtain all possible combinations of the Pythagorean triads.

In the programming of this problem, place the shortest leg of the triangle so that it will range from 3 to 100. By using the computer, the results were obtained in only 12 seconds. It can be seen that this rule does not solve for all values because of the limits set on a. It was most interesting to observe that rules dating from 500 B.C. and a modern computer can obtain the results in such concise form.

Problem 2 -- By developing a new rule, try to obtain the possible combinations of triads.

A searching technique program was constructed for the computer. Certain values for a and b were fed into the computer. a ranged from 3-50 and b from 4 to 1000. In the process of running the program, the computer would print all triads in combination with a, b, and c. The results were obtained for values of a from 3 through 50. An interesting aspect of this program is revealed in the determination of the triads. The amount of computer time taken for this program was 75 seconds.

It can be concluded, through experimentation with the preceding rules, that no one rule can apply in all instances. In comparing the two preceding problems, it is quite evident that the older, established rules give better, more concise results. The computer enables its user to save many man hours of work in arriving at solutions of new, as well as old, problems.

(8)

FLOW CHART
TRIADS

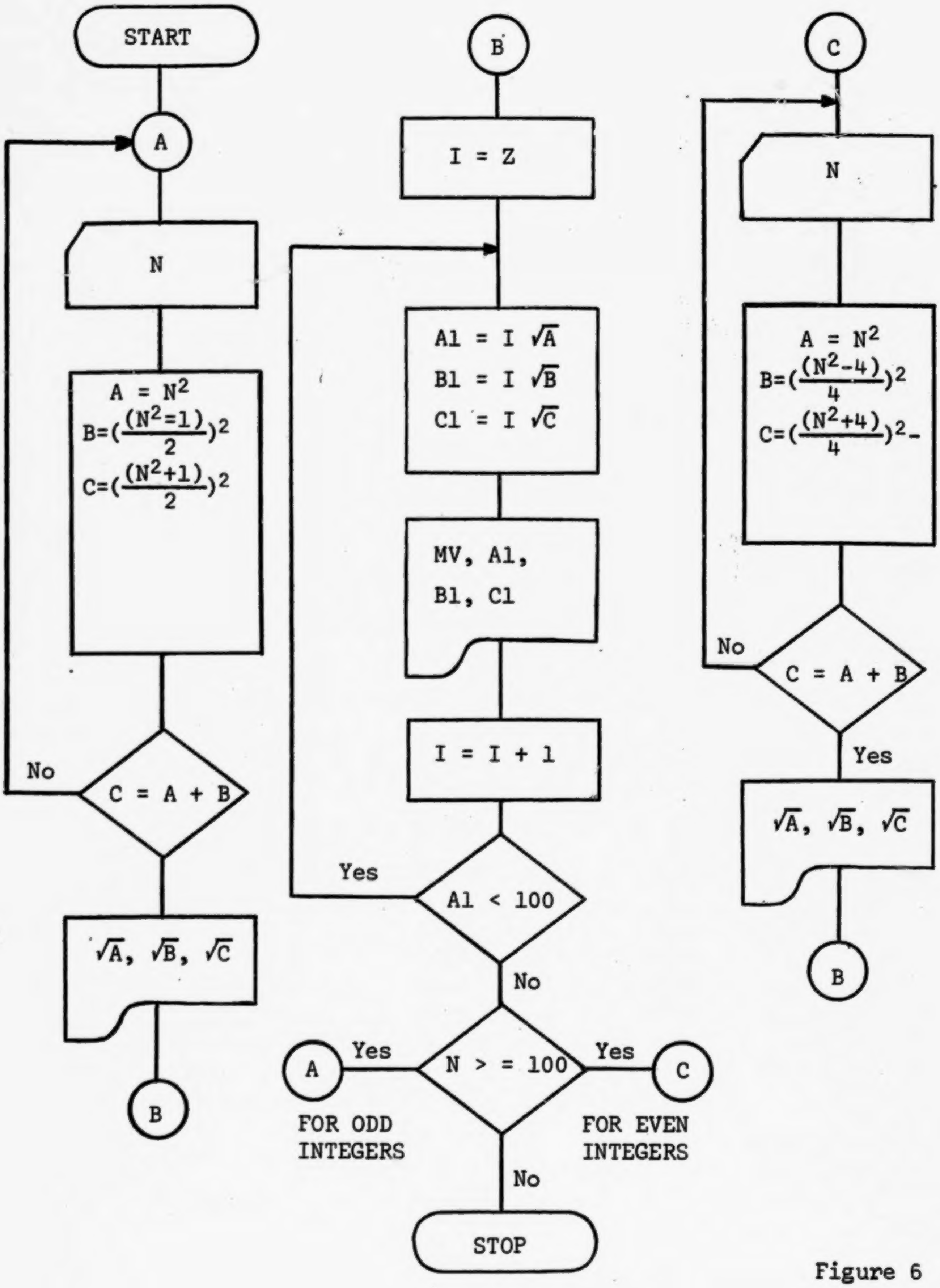

START
A
N
$A = N^2$
$B=(\frac{(N^2=1)}{2})^2$
$C=(\frac{(N^2+1)}{2})^2$
No
C = A + B
$\sqrt{A}$, $\sqrt{B}$, $\sqrt{C}$
B
B
I = Z
A1 = I $\sqrt{A}$
B1 = I $\sqrt{B}$
C1 = I $\sqrt{C}$
MV, A1,
B1, C1
I = I + 1
Yes
A1 < 100
No
A
Yes
N > = 100
Yes
C
FOR ODD INTEGERS
FOR EVEN INTEGERS
No
STOP
C
N
$A = N^2$
$B=(\frac{(N^2-4)}{4})^2$
$C=(\frac{(N^2+4)}{4})^2-$
No
C = A + B
Yes
$\sqrt{A}$, $\sqrt{B}$, $\sqrt{C}$
B

Figure 6

Problem 1

```
LISTNH
5 PRINT "A","B","C"
10 FØR N=3 TØ 99 STEP 2
20 LET A=N↑2
30 LET B=((N↑2-1)/2)↑2
40 LET C=((N↑2+1)/2)↑2
50 IF C=A+B THEN 75
70 GØ TØ 80
75 PRINT SQR(A),SQR(B),SQR(C)
76 GØSUB 220
80 NEXT N
110 FØR N=8 TØ 100 STEP 4
120 LET A=N↑2
130 LET B=((N↑2-4)/4)↑2
140 LET C=((N↑2+4)/4)↑2
150 IF C=A+B THEN 175
170 GØ TØ 180
175 PRINT SQR(A),SQR(B),SQR(C)
176 GØSUB 220
180 NEXT N
220 LET I-2
230 LET A1-I*SQR(A)
240 LET B1-I*SQR(B)
250 LET C1-I*SQR(C)
260 PRINT "MV"A1,B1,C1
270 LET I=I+1
280 IF A1<=100 THEN 230
290 RETURN
350 END

RUN


TRIADS      11:07      PX WED Ø7/12/67

A                B                C
 3                4                5
MV 6             8                10
MV 9             12               15
MV 12            16               20
MV 15            20               25
MV 18            24               30
.....            ..               ..
MV 192           4606             4610
 100             2499             2501
MV 200           4998             5002
MV 200           4998             5002

RETURN BEFØRE GØSUB IN 290


TIME:  12 SECS.


BYE

*** ØFF AT 11:30   PX WED Ø7/12/67.
```

FLOW CHART
GUESS

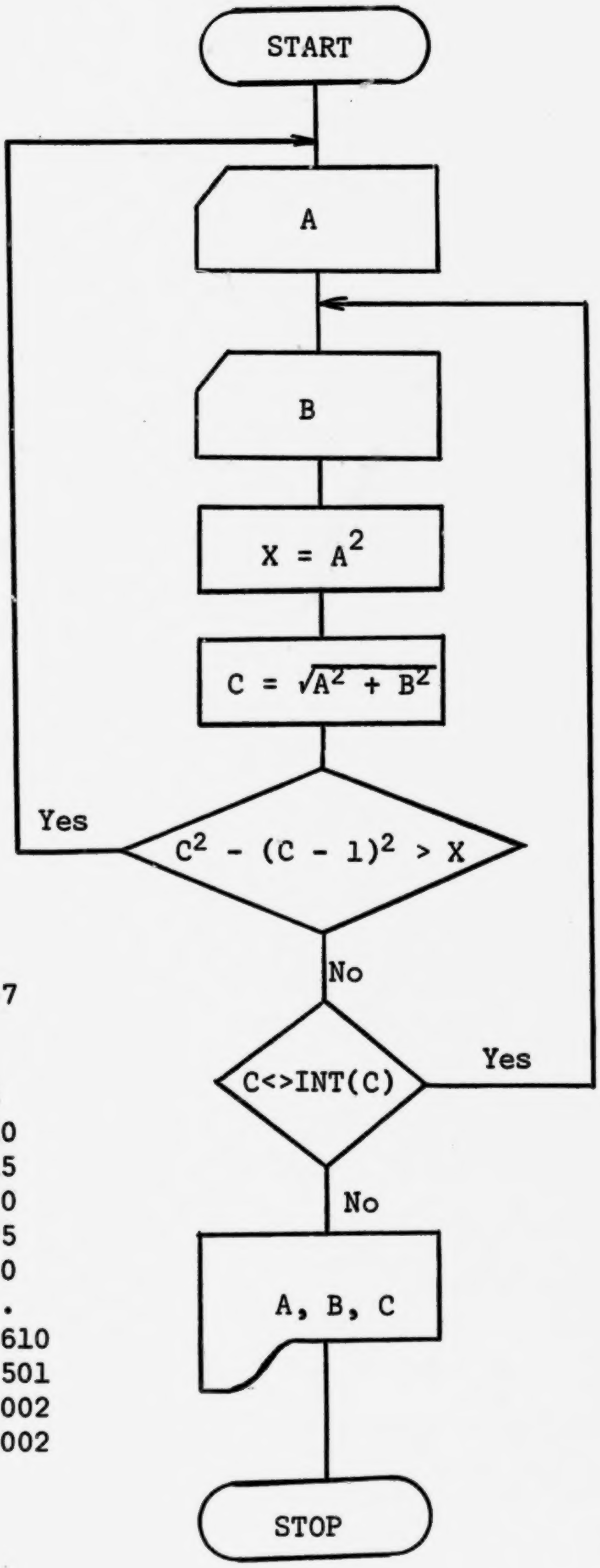


Figure 7

Problem 2

```
CU-CE-T-SHAR

EXPERIMENTAL SYSTEM LØADED UNTIL 9:30.  PLEASE LIST "EXPERI***".

ØN AT      8:27      PX FRI 07/14/67      TTY   3

USER NUMBER--P61000
SYSTEM --BAS
NEW ØR ØLD--ØLD
ØLD PRØBLEM NAME--GUESS


READY.
FINK!
LIST


GUESS      8:28      PX FRI 07/14/67

5 PRINT "A"; "B"; "C"
10 FOR A = 3 TØ 25
020 FØR B-4 TØ 1000
25 LET X = A*A
30 LET C = SQR(X + B*B)
40 IF C*C -(C-1)*(C-1)>X THEN 80
50 IF C<>INT(C) THEN 70
60 PRINT A;B;C
70 NEXT B
80 NEXT A
90 END


10 FØR A = 3 TO 100
10 FØR A = 3 TO 50
RUN


GUESS        8:29      PX FRI 07/14/67

ABC                                      TIME:  1 MINS. 15 SECS.
 3       4      5
 5       12     13
 6       8      10
 7       24     25
 8       6      10                       RAN      450/6 SEC.
 8       15     17
 .       ..     ..                       STØP.
 28      195    197                      READY.
 29      420    421
 ..      ...    ...                      UNSAVE
 48      575    577                      READY
 49      168    175
 50      120    130                      BYE

 50      624    626                      *** ØFF AT  8:36        9 MIN. CHARGE.
```

NUMERICAL ANALYSIS AND DEMONSTRATIONS FOR P.S.S.C. LABORATORY EXPERIMENTS

by Wayne B. Daniels

Numerical Analysis of P.S.S.C. Laboratory Experiment

I-1 Short Time Intervals

References

Physics Laboratory Guide, 2nd Edition; Physical Science Study Committee, D. C. Heath and Company, 1965, pp. 103.

Physics, 2nd Edition, Physical Science Study Committee, D. C. Heath and Company, 1965, pp. 12-14 "Repetitive Motions; the Stroboscope."

An objective of this laboratory experiment is to calibrate the vibrating clapper of an electric bell by determining its period of vibration. The vibrating clapper and a tape moving through the clapper will be used to study the motions of objects used in later laboratory experiments. A second objective of this experiment is to gain some appreciation for the disc-stroboscope as a simple device useful in measuring rapid periodic motions. The numerical analysis part of the experiment requires that the student actually analyze the relationship between the vibrations of the clapper and the rotations of the disc-stroboscope and express this relation in an algorithm. The learning experience gathered from the logical derivation of this algorithm and from the building of the accompanying flow chart is the primary objective of this revised experiment.

The first periodic motion measured is that of a weighted vibrating steel blade clamped to the edge of the table. The motion is slow enough to allow the observer to "stop" the motion of the end of the blade by rotating the disc-stroboscope with only one of its twelve slits open. The rate of vibration of the weighted blade is increased by adjusting the weight closer to the secured end of the blade.

The bell clapper is powered by a 1.5 volt battery and is attached to a desk top. To reduce the rate of vibration, the experimenter might weight the end of the clapper with a clothespin for the first stroboscope reading. By a trial and error process, the experimenter must rotate the stroboscope at different speeds and open the slits one by one until the motion of the clapper has apparently stopped. The student should detect a relation between the frequency of the clapper, the frequency of the stroboscope, the number of slits opened in the stroboscope, and the apparent stopping of the motion of the clapper by the rotation of the stroboscope.

The algorithm and flow chart shown in Figure 1 illustrate how the student's trial and error technique could be replaced by a computer's search for the number of slits and the frequency of the stroboscope needed to "stop" the motion of the clapper.

Let F_c = the frequency of the clapper

F_s = the frequency of the stroboscope needed to stop the motion of the clapper

n = the number of slits in the stroboscope

k = any whole integer greater than zero

Then, $(k)(F_c)=(n)(F_s)$

$$F_s = \frac{KF_c}{n}$$

Values F_c, the frequency of the clapper, and values F_s, the frequency of the stroboscope, must enter the computer as input. Since the frequency of the stroboscope depends upon hand turning, its range of frequency should extend from thirty to ninety rotations per minute. The maximum number of slits available on the stroboscope is twelve. This flow chart lets the number of slits begin at six and allows the number to range down to one and up to twelve depending upon the value of k which is established in the algorithm.

Consideration must be taken that if the frequency of the clapper is any whole integer multiple (k) greater than the frequency of the one-slit, disc-stroboscope, the stroboscope will appear to stop the motion of the vibrating clapper. In this case, however, a unique solution for the frequency of the clapper cannot be determined. If the frequency of the clapper is any unit fraction (i.e., $\frac{1}{2}$, $\frac{1}{3}$, $\frac{1}{4}$, $\frac{1}{5}$, etc.) of the frequency of the stroboscope, then the stroboscope will appear to stop the motion of the clapper at several locations of the rotating slit. Again, no unique solution for the frequency of the clapper is possible. To eliminate the possibility of such multiple frequencies, the value of (k) is tested by the computer. Should the value of (k) be less than one or greater than two, fewer or additional slits are selected ranging from one to twelve, forcing (k) within the required limits.

The computer will supply the experimenter with the frequency of the clapper being tested, the lowest frequency of the stroboscope needed to "stop" the motion of the clapper, and the number of slits needed.

Summary Questions

1. What adjustments may be made to the disc-stroboscope to allow the experimenter to measure frequencies greater than that of the stroboscope?

2. The flow chart shows an input of clapper frequencies and stroboscope frequencies. How does the output differ from the input?

3. If the computer should select more slits than are available on the stroboscope, what warning would the computer give the experimenter?

4. Under what conditions would the computer select more slits than are available on the stroboscope?

5. What provision has been made in the flow chart, Figure 1, to make sure that the computer does not match stroboscope frequencies with multiples and unit fractions of clapper frequencies?

Answers to Summary Questions

1. The number of slits opened in the stroboscope must be increased.

2. The output from the computer will list the lowest frequency and the least number of slits needed to match the frequency of the clapper.

3. The computer will print out "upper limit," and then proceed to the next clapper frequency.

4. The computer would select more slits than are available on the stroboscope if the clapper were vibrating with a frequency greater than twelve times the maximum frequency of the stroboscope.

5. Two logic boxes located in the flow chart test (k) for a value greater than two or less than one. If (k) is greater than two or equal to two, the number of slits is increased by one. If (k) is less than one, the number of slits is reduced by one. When $1 \leq k > 2$, F_s is multiplied by (k) to find the lowest stroboscope frequency and the least number of slits needed to "stop" the motion of the clapper.

Numerical Demonstrations for P.S.S.C. Laboratory Experiments

II-11 Waves from Two Point Sources

II-12 Interference and Phase

II-13 Young's Experiment

References

Physics Laboratory Guide, 2nd Edition; Physical Science Study Committee; D. C. Heath and Company, 1965, pp. 28-31

Physics, 2nd Edition; Physical Science Study Committee; D. C. Heath and Company, 1965, Chapter 17, "Interference," pp. 282-292.

The main purposes of the three experiments are to provide a situation where the experimenter can observe the behavior of waves generated at two point sources in a ripple tank and apply the resulting conclusions to light waves passing through two slits in a coated glass slide. Waves generated at two point sources will interfere with one another, producing a pattern of nodal lines whose changes are the primary concern of this experiment. The angle that a given nodal line makes with the line connecting the two point sources is a function of the wavelength of the wave, the distance between the point sources, and the phase delay of the waves at the second source. The purpose of the numerical demonstration of this experiment is to show how the angle of a nodal line changes as values of wavelength, distance between point sources, and phase delay change.

Figures 2 and 3 illustrate the mathematical derivation of the algorithm, showing the dependence of the angle of the nodal line.

Let P_n = a point located on the nodal line (n).

x = the distance of the point from the bisector.

θ = the angle formed by the bisector and a line connecting the midpoint between the point sources and point P_n.

d = the separation between the point sources.

λ = the wave length of the waves at both point sources.

p = the phase delay or that fractional part of the wave's period by which it is delayed.

$p\lambda$ = the distance by which one wave is delayed compared to the other.

n = the number of the nodal line right or left of the bisector.

From Figure 3

$$PS_1P_n - PS_2P_n = (\sin\theta)(d) = \frac{x}{P_d}$$

$$\sin\theta = \frac{(n - \frac{1}{2} + p)\lambda}{d}$$

The flow chart, Figure 4, shows the manner by which the angle θ shall change. Nodal line (two) is selected at random for demonstration purposes. As the distance (d) between the point sources is decreased from four centimeters to one centimeter by steps of one centimeter, the phase delay is lengthened from zero to one by steps of 0.2, and the wave length is increased from one to three centimeters by steps of one centimeter. Sin θ is printed corresponding to each combination of d, p, and λ.

Summary Questions

1. Which variable in the algorithm is instructed by the flow chart to change first?

2. How many values of sin θ would be printed by the instructions of this flow chart?

3. Rewrite the flow chart in a way in which it would demonstrate whether or not there is equal angular spacing between nodal lines when the variables are held constant.

Answers to Summary Questions

1. That variable read-in by the inner loop of the flow chart changes first. It is the variable (d).

2. Seventy-two.

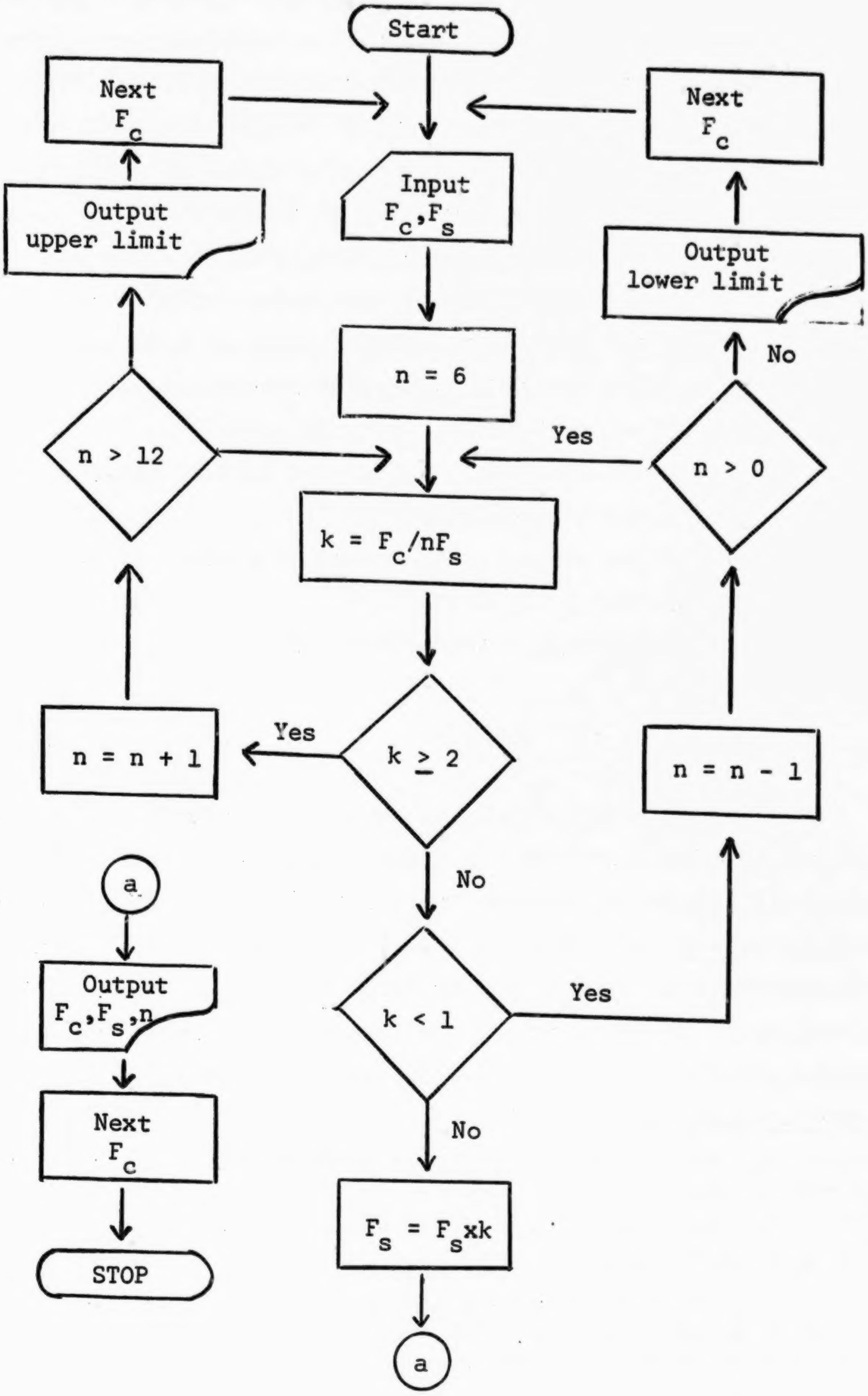

Start
Input F_c,F_s
n = 6
k = F_c/nF_s
k ≥ 2
Yes
n = n + 1
n > 12
Output upper limit
Next F_c
No
k < 1
Yes
n = n - 1
n > 0
Yes
No
Output lower limit
Next F_c
No
F_s = $F_s$xk
a
a
Output F_c,F_s,n
Next F_c
STOP

Figure 1

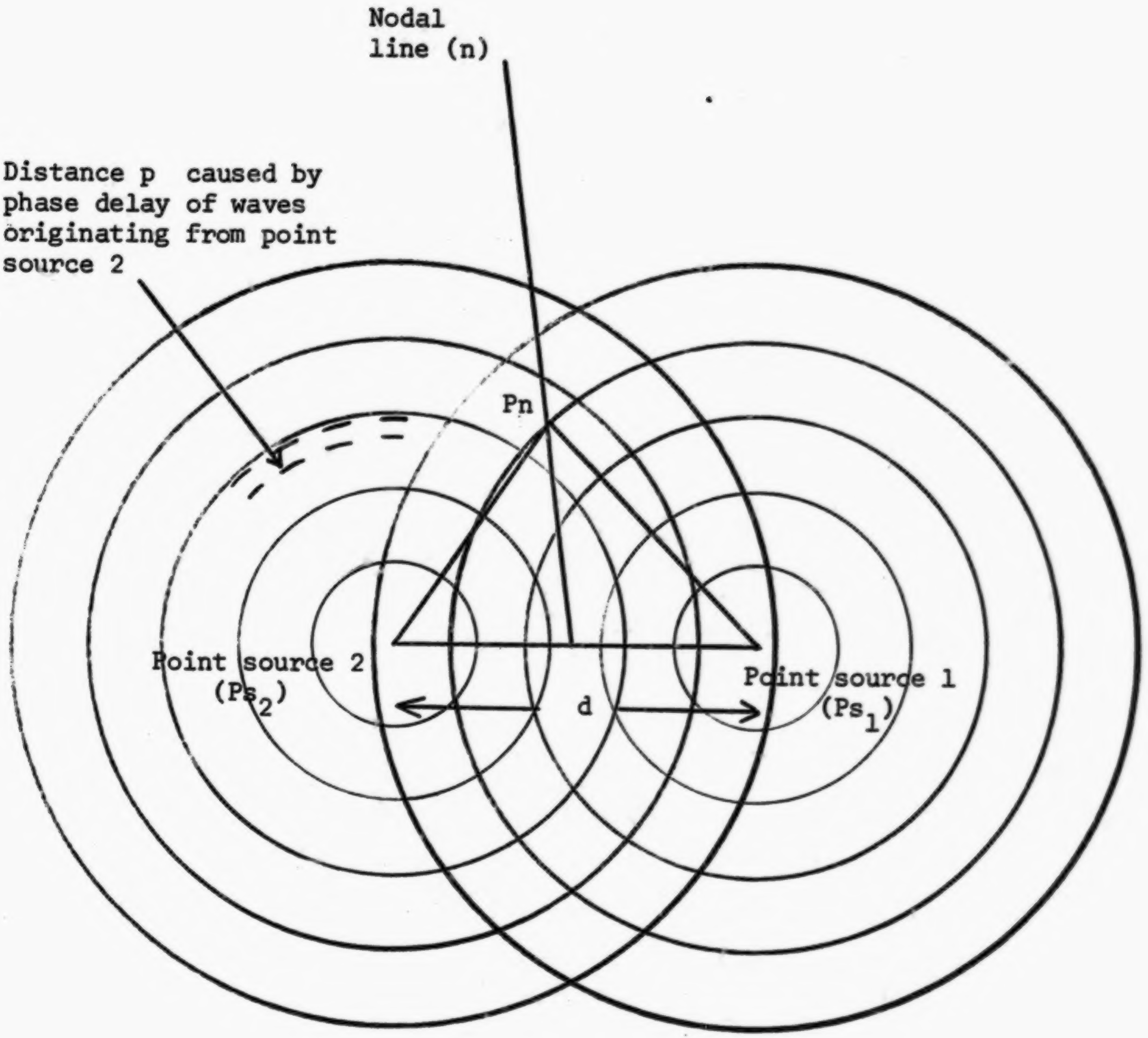

Nodal
line (n)
Distance p caused by
phase delay of waves
originating from point
source 2
Pn
Point source 2
(Ps_2)
d
Point source 1
(Ps_1)

Figure 2

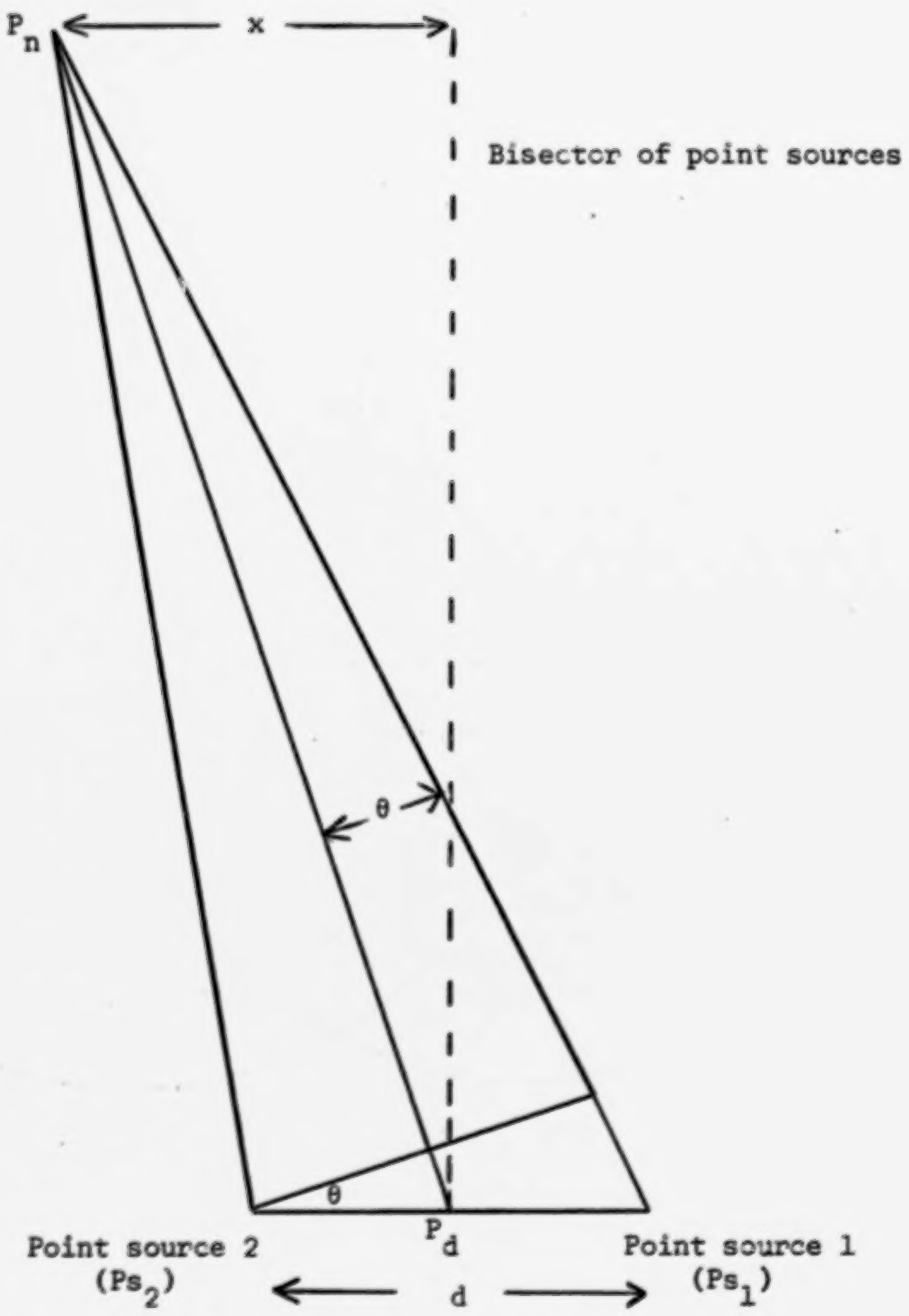

P_n
x
Bisector of point sources
θ
θ
P_d
Point source 2
(Ps_2)
d
Point source 1
(Ps_1)

Figure 3

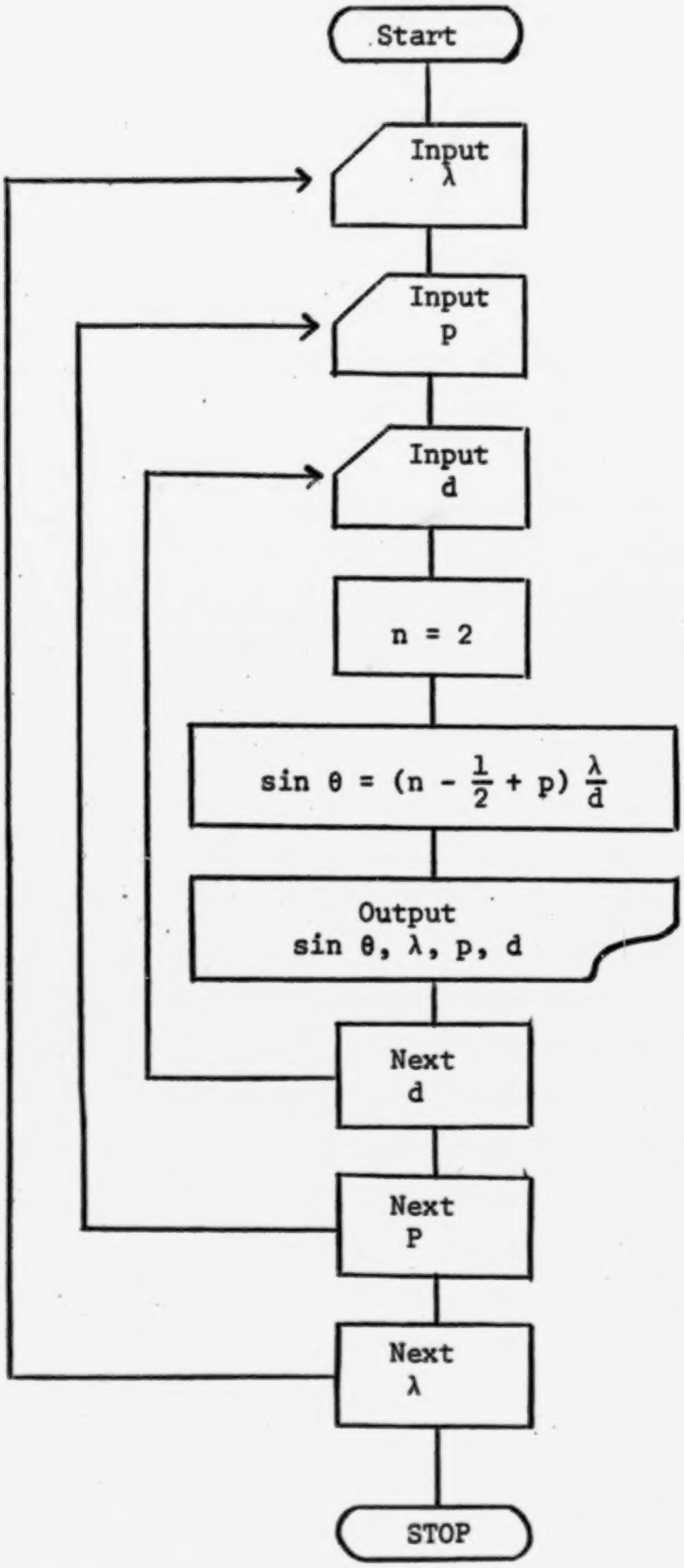

Start
Input
λ
Input
p
Input
d
n = 2
sin θ = (n - 1/2 + p) λ/d
Output
sin θ, λ, p, d
Next
d
Next
P
Next
λ
STOP

Figure 4

SOME USES OF THE COMPUTER IN PHYSICS TEACHING

by William E. Einert and Hepburn Ingham

Since computers can do prodigious amounts of arithmetic with great speed and accuracy, they have important application in solving some physics problems. In physics the attempt is made to understand nature in a quantitative way. A physical variable is related to another variable or a constant in a precise way. For a value of x one tries to determine by suitable arithmetic the value of f(x). One can work with a known functional relationship to determine f(x), or can work with data to determine what the functional relationship is.

Many topics taught in the customary high school physics course could be augmented, enriched, or more easily understood by computer techniques. The purpose of this paper is to identify a few of these topics and to show how they might be introduced and taught through the use of the computer and allied techniques.

Of equal importance is the gain which can be made by the student in learning about computer methods and in establishing foundational understanding on the use of computers. In other words, here the computer is actually used as an adjunct to problem solving and an aid to understanding. It is not taken in isolation (as it would be in a separate course) or as an interesting, novel "gimmick" (as it might be in a mathematics class).

The following topics are presented mostly as hints to the knowledgeable teacher. They are not meant as finished teaching units, but could serve as curriculum guide lines for the teacher in presenting and utilizing each concept. Most of the material could be given to the student in printed form as it is written here if it is accompanied by a well planned lecture, class discussion, laboratory experience, and appropriate exercises. The material is written with the assumption that the student has had an introduction to BASIC language and some experience with flow charting.

An interesting set of physical relationships is met early in the P.S.S.C. physics. These exist on the subject of scaling. Intuitively most students believe that, for any physical structure, the full-sized object will have all of the properties of a similar miniature model. That is, they are of the same material; and the large structure is simply increased by some constant in every dimension over the small model. This belief turns out to be surprisingly false. If the dimensions of the model increase over a given range, the relationship of support strength to weight varies widely for an increase in dimension (R). This is shown by the graph in Figure 1. The support strength is directly proportional to the cross-sectional area of the supports for the structure, which is related to (R^2). On the other hand the weight is directly related to the volume, which is related to (R^3). One can see that the weight increases much more rapidly than support strength for an increase in dimension (R).

An illuminating exercise would be to increment (R) in unit steps on the computer and to get the strength-to-weight ratio. This ratio would decrease to some critical value (Q), an allowed safety factor, and would give some idea of maximum size of the structure. The problem could be extended and (R) allowed to increase to where the strength-to-weight ratio equals unity, at which the structure would fail. The flow chart for the program could be as shown in Figure 2. The parameters in the program are defined as follows:

R = dimension of the structure

K_1 = a constant relating dimension (R) to crossectional area or support strength

K_2 = a constant, relating dimension (R) to volume and weight of the structure

Q = a safety factor, a minimum safe ratio of strength to weight

A problem that the student might like to consider is that of a cylindrical support column whose material has x lbs/in^2 compressional strength. The value of (x) for various materials can be found in a handbook along with the density (ρ) of the material. The relationships would be:

$$\text{Strength} \propto \text{Area} = (R\ \text{in})^2\ (x\ \text{lbs/in}^2)$$

$$\text{and } K_1 = (\pi)(x\ \text{lbs/in}^2)$$

$$\text{Weight} \propto \text{Volume} = (\pi R^2 h\ \text{in}^3)(\rho\ \text{lbs/in}^3)$$

$$\text{and } K_2 = \pi\rho\ \text{lbs/in}^3$$

A general area in which computers do valuable work is in the handling of experimental data. If one is looking for the function that exists between two variables it is desirable to have some aid to do arithmetic. This is especially true where a large number of data values are gathered in a short time. A specific example occurs in the refraction of light. Here the student can rapidly measure many angles of incidence and refraction (θ_i) and (θ_r) respectively. In searching for the functional relation that exists between these variables, the student needs to do such operations as (θ_i/θ_r), ($\theta_i \times \theta_r$), ($\sin\theta_i/\sin\theta_r$), etc. If a computer could be utilized the student could be encouraged to take many more data measurements in limited laboratory time.

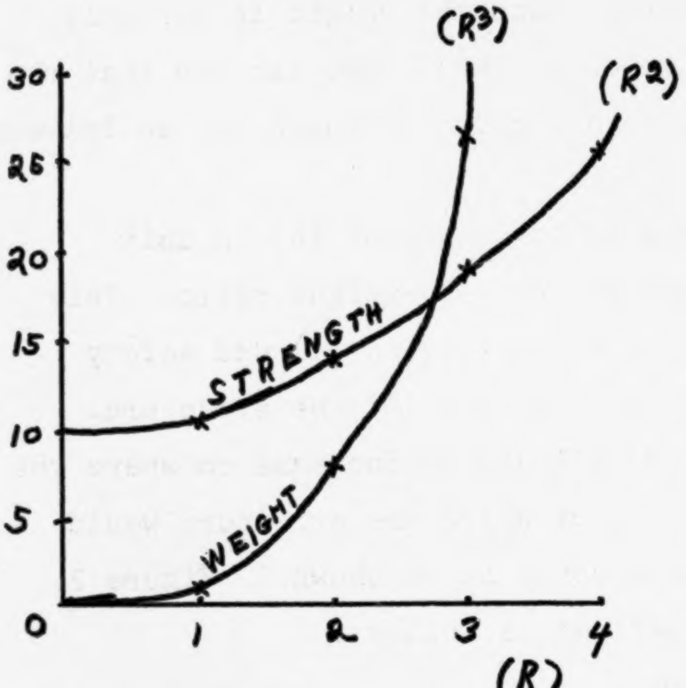


FiG. 1

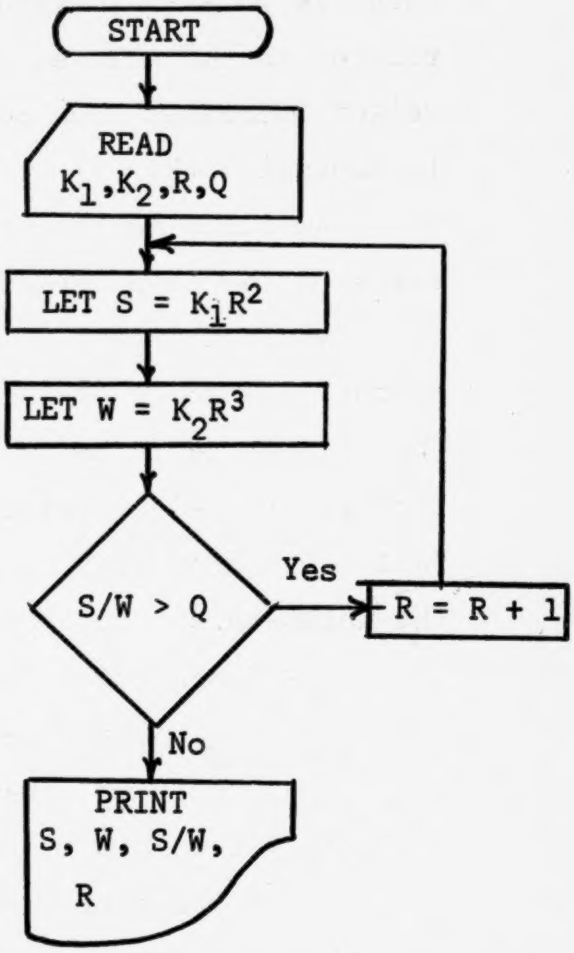

START
READ
K1,K2,R,Q
LET S = K1R²
LET W = K2R³
S/W > Q
Yes
R = R + 1
No
PRINT
S, W, S/W,
R

FiG. 2

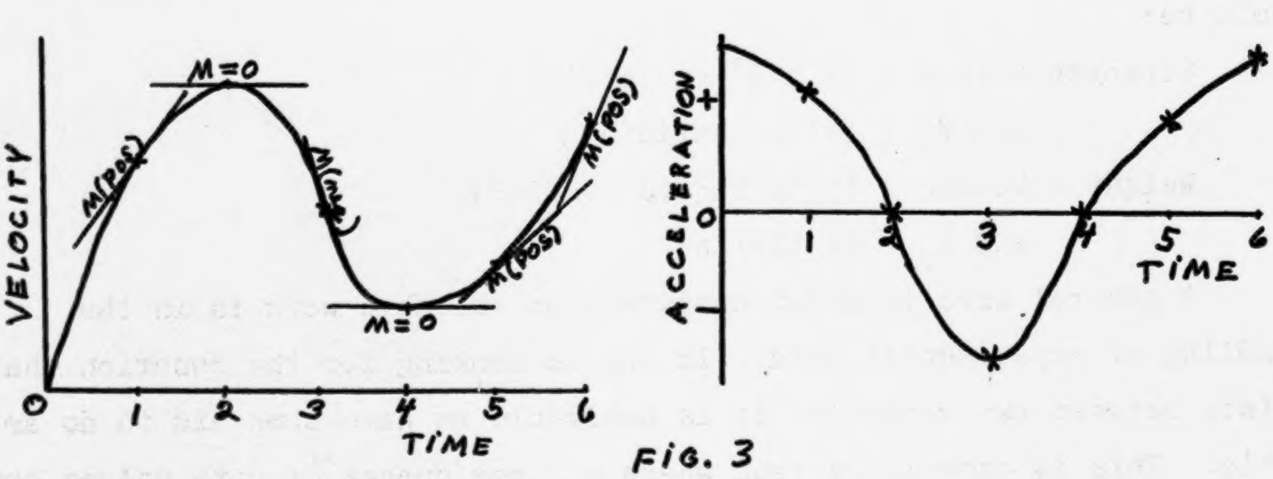


FiG. 3

Another problem in physics is getting the slopes of curves at several points. For example, taking the slope of a velocity-time graph to get the rate of change of velocity. By definition this is the acceleration. One could reproduce an acceleration graph from a velocity-time graph as illustrated in Figure 3.

The computer could be programmed to compute the slopes at a large number of points. By so-doing, the student could get a good feeling for the significance of slope and could make a much more accurate representation of the acceleration-time graph.

An important mathematical function in physics is the inverse square relationship. This is encountered when studying the effects of radiation, such as light, on a surface at some distance, (x), from a point source. The function also applies to electrical and gravitational force fields, which act from a static charge center or a gravitational center. In general, the relationship can be expressed as $y \propto 1/x^2$. In the case of radiation, the intensity (I) on a surface at a distance (R) from a point source is given by: $I = \text{constant}/R^2$. The functions for the electric and gravitational force fields are defined by $F = KQ_1Q_2/R^2$ and $F = GM_1M_2/R^2$. In a given problem the terms KQ_1Q_2 and GM_1M_2 can be constant, and a computer used to find the force (F) for any separation of the centers (R). A flow chart for a gravitational problem could be as in Figure 4.

In a satellite problem (see Figure 5) the masses of the satellite, earth, and the universal gravitational constant combine in the equation to be GM_1M_2. For a measured or given set of radii, $R_1, R_2, \ldots R_n$, the corresponding forces $F_1, F_2, \ldots F_n$ can be calculated. In this case the gravitational force provides the necessary centripetal deflection. Knowing the centripetal force, satellite mass, and radius one can also calculate the velocity and time of revolution of the satellite. A consistent set of units must be used.

Calculation of the total resistance of parallel electric circuits would very quickly bring the student to see the effectiveness of the computer. The reciprocal relationships $\frac{1}{R_t} = \sum_{i=1}^{n} \frac{1}{R_i}$ become very burdensome handwork when the r's are large or non-integer, or when n is large. The summation of these unit fractions is customarily tackled by the student as if they were ordinary fractions. The student often laboriously finds a common denominator and proceeds in that fashion.

A series of such problems, starting with small integer values for r and proceeding to become more computationally lengthy, should quickly bring the student to his knees begging for computer help.

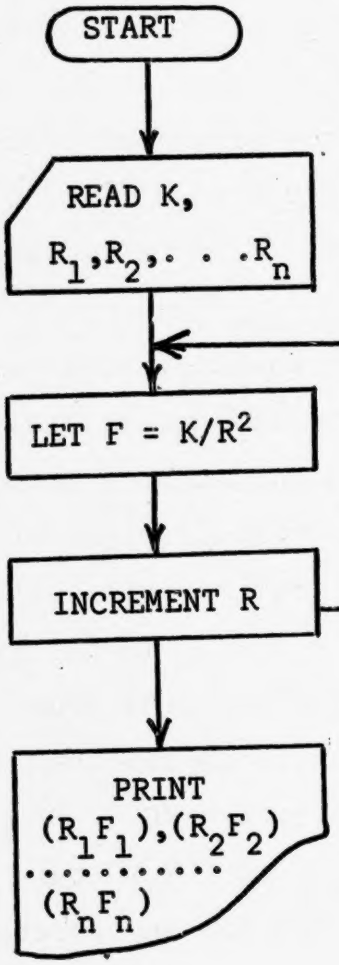


FIG. 4

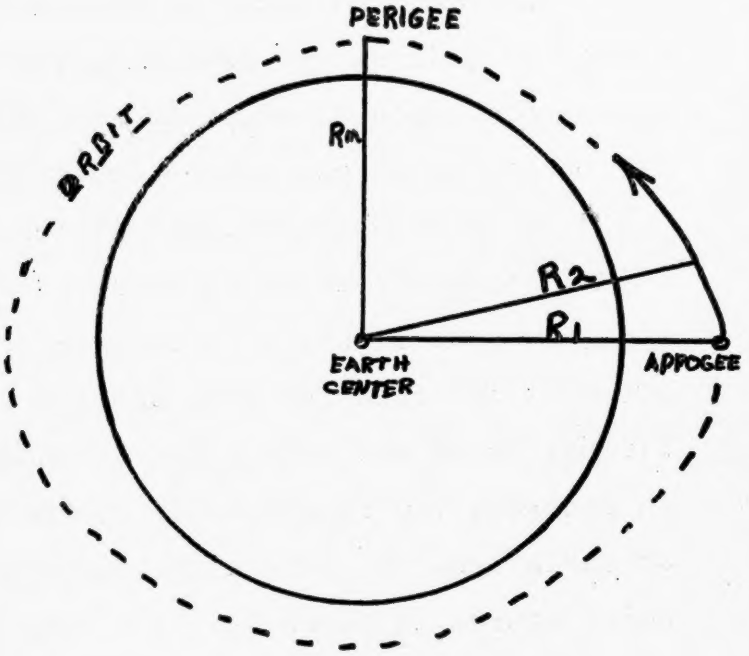


FIG. 5

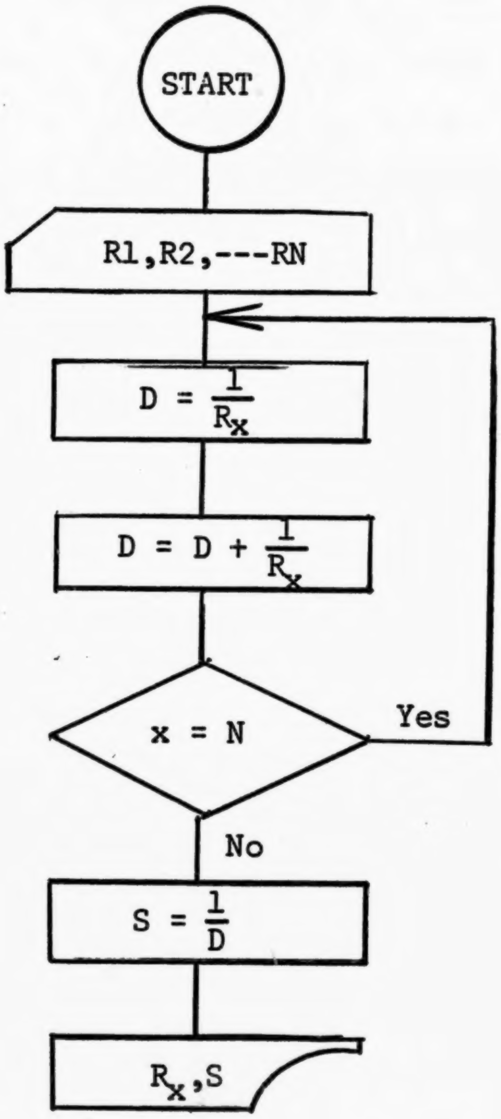
When the student wants the computer he is more likely to appreciate it, and the motivational level for learning is greater.

In addition, the availability of a computer would make work with more complex circuitry possible. These are generally enjoyable to the student and can serve as springboards into other concepts and topics.

Even if hardware computer is not available, the software of the flow chart and program for this would be illuminating. See Figure 6 for the Flow Chart and Program Notes.

Finding the area under curves (i.e., the measure of the interior of the figure bounded by $y = f(x)$, $x = a$, $x = b$, and $y = 0$) comes up in the study of distance, velocity, acceleration, and time inter-relationships. Customarily this problem is solved by the rectangular step method. If many steps are used (a small increment selected), or if the function under discussion is non-linear, the arithmetic work quickly becomes quite lengthy. For this reason, students (and teachers) never extend their work to the place where the results would be considered satisfactory. Furthermore, the student is apt to miss the whole basic

Flow Chart



Program

```
 10  READ N
 20  For x = 1 to N
 30  READ R(x)
 40  LET D - D + 1/R(x)
 50  NEXT x
 60  LET S - 1/D
 70  PRINT S
 80  DATA 5, 2, 3, 4, 5, 6
 90  DATA 8, 2, 4, 6, 8, 10, 12, 14, 16
100  TO TO 10
110  END
```

Notes

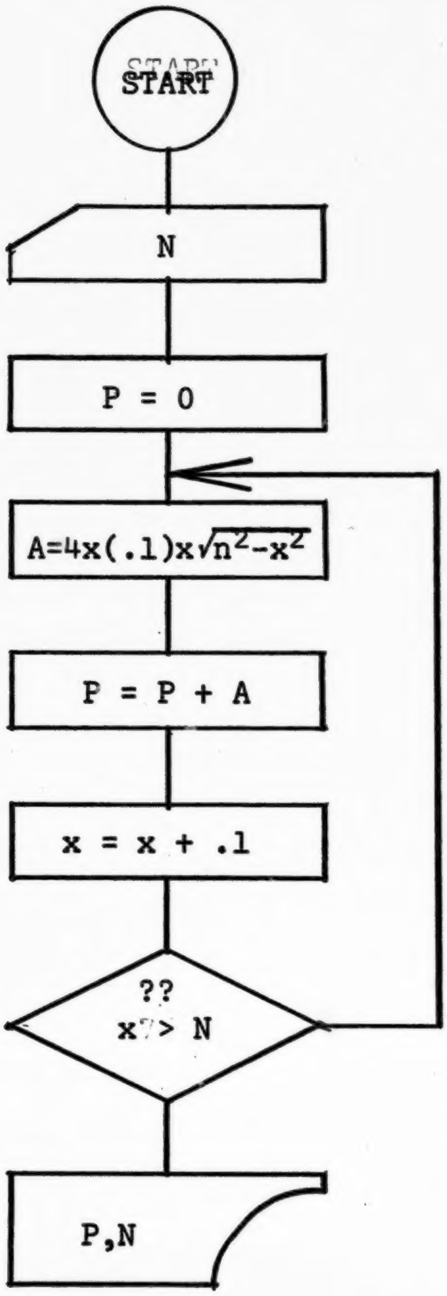
Line 10 reads 5 into the machine. Lines 20 and 30 pick up the resistances 2, 3, 4, 5, and 6 from the data in line 80. Lines 40 through 60 perform the actual calculations. Line 100 tells the computer to go back and pick up a new set of data.

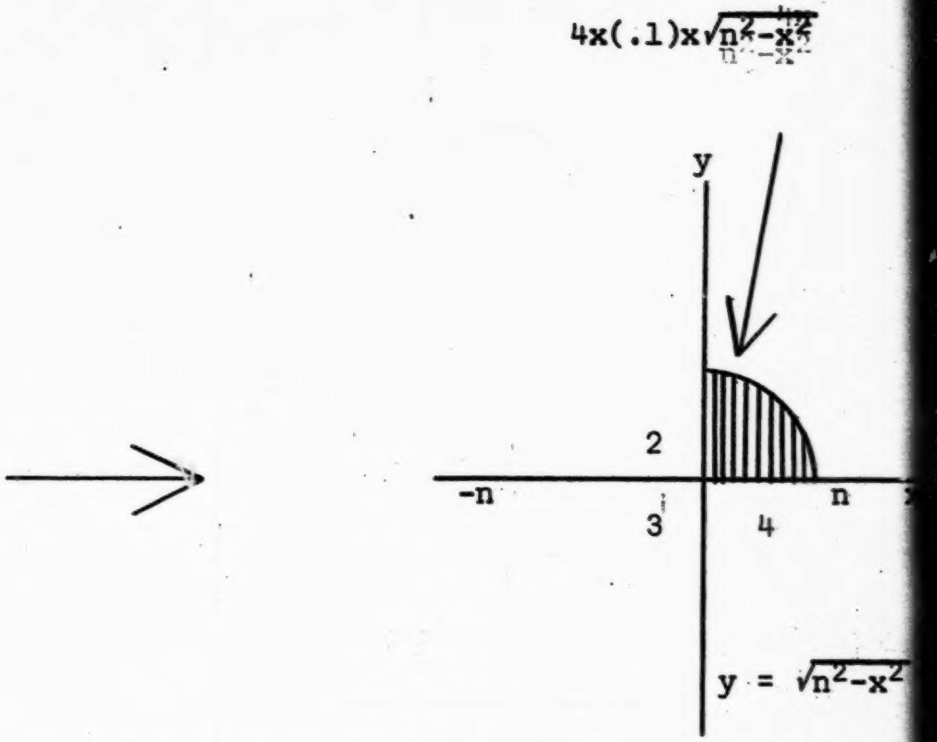
Figure 6

concept of setting an increment which approaches a limit -- and then is denied a beautiful insight into the concept and workings of integral calculus.

In the absence of hardware, the flow chart alone would be of some benefit in bringing about some of the objectives. Students should be allowed to construct their own flow charts and should be encouraged to build in a loop which runs the program through with successively smaller increments (greater number of rectangle steps). This would tend to promote a feeling for convergence to the current value. See Figure 7.

A computer program for projectile motion could be a useful tool in studying some of the important relationships in kinematics. Given various muzzle velocities and angles we can compute the missile's time in air, its maximum height, and its range. The calculation for such a problem can be very time-consuming, which detracts from gaining overall knowledge of the kinematics involved. This is particularly true if





Sector 1 is divided into rectangles of width .1 and length equal to the functional value at incremented x values. 4 times the summation of these rectangles would give the total area of the circle.

Figure 7

we want to make numerical comparisons for a number of muzzle velocities and angles. Included in this particular program are: (1) the problem of converting from degrees to radian measure (the computer uses only radian measure); (2) the vector relationship of the muzzle velocity to its vertical and horizontal components -- incorporated in the program by use of the sine and cosine functions. The computer program is straight forward and probably of more value than a flow chart. Included are notes with the program (see Figure 8) explaining the various variables and ideas involved.

Projectile Program

```
100 READ V, N, M                    (V IS MUZZLE VELOCITY, M, N FRACTIONAL
110 LET A=N*3.14/M                  PARTS OF Π TO CHANGE FROM DEGREES TO
120 LET V2=V*SIN(A)                 RADIANS IN LINE 110)
130 LET V1=V*COS(A)
                                    120, 130 GET HORIZONTAL AND VERTICAL
140 LET T1-V2/32
                                    VECTOR COMPONENTS OF V
150 LET T=2*T1
                                    140-170 ARE THE MOTION FORMULAE
160 LET D=V1*T
170 LET H-16*T1*T1
180 PRINT "V2="V2, "V1="V1, "T="T, "D="D, "H="H, "A="A,
190 PRINT "V="V
200 GO TO 100
210 DATA 1000, 1, 4, 500, 1, 4, 1000, 1, 2, 1000, 2, 3, 1000, 0, 2, 1000,
220 DATA 2, 0,
230 END

RUN
WAIT.

HEP12*     14:23     PX TUE 07/18/67

V2= 706.825      V1= 707.388      T- 44.1766      D= 31250        H- 7806.28
A= .785          V= 1000
V2= 353.413      V1= 353.694      T= 22.0883      D= 78125        H= 1951.57
A= .785          V= 500
V2= 1000.        V1-0             T= 62.5         D=  0           H= 15625.
A= 1.57          V= 1000
V2= 866.556      V1=-499.08       T= 54.1597      D=-27063.3      H= 11733.1
A= 2.09333       V= 1000
V2= 0            V1= 1000         T= 0            D=  0           H= 0
A= 0             V= 1000

V2=-4.65661 E-7                   V1=-4.65661 E-7                 T=-2.91038
D=2 0            H= 3.38813 E-15                  A= 5.78960 E 76
V= 1000
```

Figure 8

FACTORS G.C.F. PRIME FACTORIZATIONS

By Sandra W. Epstein and Cheryl Gaskell

Introduction to Teacher

a) TOPIC: G.C.F. approached through prime factorization, factors and Euclid's Algorithm.

b) LEVEL OF EXERCISE: Seventh grade through Algebra I.

c) BASIC KNOWLEDGE NEEDED:

*ability to write a simple program with flow chart.

*commands such as LET, PRINT, GO TO, READ, END SQR(A), INT(A), FOR-NEXT.

d) SUGGESTED INSERTION INTO CLASS WORK: after study of primes, factors, and G.C.F. has been covered in class.

e) APPROACH USED: The discovery approach is suggested for this unit. By asking the correct questions and showing examples, the teacher should try to guide the students into a knowledge of what they need to know and need to change. To find the G.C.F. we have approached the program through a knowledge of factors and have finished by using Euclid's Algorithm. Given as supplementary topics are the programs for finding prime factorization of any number and finding the L.C.M. for any two positive integers.

I. PROBLEM: To find the greatest common factor of any two positive integers A and B using both the factor approach and Euclid's Algorithm approach. This is designed to be a class project to write and rewrite the programs together. Also included at the end are the flow charts, example boxes, suggested homework assignments and quizzes.

The class might be asked how they would program a simple division problem, A/B. They should be encouraged to experiment with a trial and error approach. They might start thus:

PROGRAM I:

```
                                   SAN1      11:21    PX TUE 08/22/67
10 READ A,B
20 LET Q=A/B                       2.5
30 PRINT Q                         7
40 GØ TØ 10                        2.33333
50 DATA 5,2,7,1,7,3                ØUT ØF DATA IN  10
60 END                             TIME:   0 SECS.
```

At this point the teacher might put example boxes on the board for A,B, and Q, and show what the computer will find. (Fig. 1) Someone should discover that the computer automatically calculates out to 6 digits. We would like to have the whole number and the remainder. Here the teacher should explain the integer function of the machine. Then the students should start again, modifying program 1. Discussion on how to find the remainder of a division problem should result in step 30.

PROGRAM II:

```
10 READ A,B                 SAN 2     11:29     PX TUE 08/22/67
20 LET Q=INT(A/B)
30 LET R=A-Q*B              1                   3
40 PRINT R,Q                1                   3
50 GØ TØ 10                 3                   1
60 DATA 10,3,7,2,8,5        ØUT ØF DATA IN      10
70 END                      TIME:     0 SECS.
```

Again the teacher should put boxes on the board (Figure 2) and show what the computer will do with this program. At this point the teacher could ask how to find factors of a number. If B is a factor of A , the remainder of the division A/B is zero. If you wanted to find all factors of A , you could start with B= 1 and check all integers up to A . Any whose remainder is zero should be printed out. The students should either be familiar with or be encouraged to think of how to get the B's to increase by one each time.

```
10 LET B=1
20 PRINT B
30 LET B=B+1
40 GO TO 20
50 END
```

NOTE: in step 30, B+1 is now the new value of B . This is not the same as B+1 = B

Combine the above with program 2 and add a check for R=0. (Step 50) You should have a program for finding factors of A .

PROGRAM III.

```
10 READ A
20 LET B=1
30 LET Q=INT(B/A)
40 LET R=A-B*Q
50 IF R<>0 THEN 70
60 PRINT B,Q,R
70 LET B=B+1
80 GO TO 30
90 DATA 12
100 END
```

NOTE: to denote 'not equal to' in computer language it is necessary to use < and > in that order.

By use of examples on the board (Figure 3) the students may note that this program finds all the factors but continues on checking the next larger B. This is called an infinite loop. We need a decision box to shut B off when it gets to be the value of A . (Step 80 below)

Program IV is adequate for finding factors of one A . For finding factors of several numbers, insert this step:

```
95 GO TO 10
```

and change step 80 to read:

```
80 IF B>A THEN 95
```

PROGRAM IV:

```
10 READ A                      SAN4     11:35   PX TUE 08/22/67
20 LET B=1
30 LET Q=INT(A/B)              1               12
40 LET R=A-B*Q                 2               6
50 IF R<>0 THEN 70             3               4
60 PRINT B,Q                   4               3
70 LET B=B+1                   6               2
80 IF B>A THEN 110             12              1
90 GØ TØ 30
100 DATA 12
110 END                        TIME:   0 SECS.
```

When the program is out of data it will tell you so. Note also in program 4 that by printing B and Q we got a list of factors twice. The factors may be found by checking from B to SQR(A) and printing out B and the quotient Q. This will give you all the pairs of factors with no repetition as you found in Figure 4. The modified program will be:

PROGRAM V:

```
                                         10 READ A
SAN5    11:42   PX TUE 08/22/67          20 LET B=1
                                         30 LET Q=INT(A/B)
1              12                        40 LET R=A-B*Q
2              6                         50 IF R<>0 THEN 70
3              4                         60 PRINT B,Q
                                         70 LET B=B+1
ØUT ØF DATA IN  10                       80 IF B>SQU(A) THEN 100
                                         90 GØ TØ 30
                                         100 GØ TØ 10
                                         110 DATA 12
TIME:   0 SECS.                          120 END
```

The last change to be suggested for finding factors of a number is the FOR, NEXT statement, thus eliminating the decision box which is step 80 in program 5. The change would be step 20 to read:

20 FOR B=1 TO SQR(A)

and step 70 would be changed to:

70 NEXT B

Step 80 and 90 would be left out. (Flow chart given at end, Figure 5)

PROGRAM VI:

```
10 READ A                      SAN6    11:47    PX TUE 08/22/67
20 FØR B=1 TØ SQR(A)
30 LET Q=INT(A/B)              1               12
40 LET R=A-B*Q                 2               6
50 IF R<>0 THEN 70             3               4
60 PRINT B,Q
70 NEXT B                      ØUT ØF DATA IN  10
80 GØ TØ 10
90 DATA 12
100 END                        TIME:   0 SECS.
```

(The next section could be assigned as homework).

You could now ask the students if, with prgram 6, they could find the

G.C.F. of two positive integers. They should see that they could enter two numbers in the data and look at the factors of each. Then find the common factors and simply take the largest. Then ask the students if they could find the G.C.F. a little more efficiently by using the division process explained in Euclid's Algorithm. With a couple of examples on the board, the teacher could point out what changes are needed in program 6 to find the G.C.F. by this second approach. We need to read in both A and B in this approach. Therefore B isn't going to be incrementing and we eliminate two steps. In the second division, the divisor B is changed to be the remainder (step 60) and the dividend A is changed to be the divisor B. (Step 50) This repeats until a zero remainder is found. When the remainder is zero, the last divisor B is the G.C.F. (Step 80), (Figure 6).

NOTE: that if A < B the computer will go through one more cycle to divide and end by placing A=B and B=R thus switching the order.

PROGRAM VII:

```
10 READ A,B                 SAN7      11:53    PX TUE 08/22/67
20 LET Q=INT(A/B)
30 LET R=A-Q*B              4
40 IF R=0 THEN 80           1
50 LET A=B                  6
60 LET B=R
70 GØ TØ 20                 ØUT ØF DATA IN 10
80 PRINT B
90 GØ TØ 10
100 DATA 12,16,5,8,48,30
110 END                     TIME:    0 SECS.
```

You may want to make the print step a little more detailed and insert a line:

```
75 PRINT "THE GREATEST COMMON FACTOR= "
```

You may also want a copy of your original A and B . To get this you would have to insert a step 15:

```
15 PRINT A,B
```

The flow chart for this program is given in Figure 7.

II. SUGGESTED HOMEWORK:

1) Write the program for G.C.F. using Euclid's Algorithm.

2) After finishing the G.C.F. orally in class, the teacher could have the students write a program for L.C.M. of any two positive integers.

3) Write a modified program to print all the factors of 48 and 72 and also to print "the greatest common factor of 48 and 72=", and then print the G.C.F. (this would combine program 6 and program 7 and could be assigned after the entire seven programs have been covered in class.)

III. SUGGESTED QUIZ:

1) Write a program using the integer function to find the greatest integer $\leq$ the given numbers:

a) 8.312

b) -4.9

c) 7.6001

2) Write a program to find the squares of the first five primes.

3) Write a program to divide 85 by all the numbers 5 through 12 and print out the quotient and the remainder in each case. (Use a FOR-NEXT statement in your program).

4) Write a program to find the set of multiples of any number A . Use A=9, 13, 25. (NOTE: the set of multiples is infinite so find at least ten multiples and then print the three dots.)

IV. SUPPLEMENTARY PROGRAM FOR L.C.M. of two positive integers. This exercise could be used as supplementary material or could be used at a later time. (The student needs to understand the DIM statement and subscripted variables).

PROGRAM VIII:

```
LISTNH
10 DIM M(200), N(200)
20 READ N,M
30 FØR K=1 TØ 100
40 LET N(K+1)=N*K
50 NEXT K
60 FØR J=1 TØ 100
70 LET M(J+1)=M*J
80 NEXT J
90 FØR I=1 TØ 100
100 FØR S=1 TØ 100
110 IF N(I)=M(S) THEN 155
120 NEXT S
140 NEXT I
150 PRINT "NØ CØMMØN MULTIPLE YET"
155 LET Y=M(S)
160 PRINT N,M,"LCM=",Y
170 GØ TØ 20
180 DATA 12,16,130,25,52,104,3,72
190 END
SAN23          10:39   PX THU 08/24/67
12             16            LCM=          48
130            25            LCM=          650
52             104           LCM=          104
3              72            LCM=          72
ØUT ØF DATA IN  20
```

V. SUPPLEMENTARY PROGRAM FOR FINDING THE PRIME FACTORIZATION OF ANY GIVEN POSITIVE INTEGER: It is assumed that the students have an understanding of prime numbers and factorization. The teacher can then use this program to introduce prime factorization. It is suggested that this take two class periods.

The first period the teacher should introduce the concept of prime factorization. The following are questions suggested for a teacher to use to begin the class discussion. First ask, "What is meant if we are asked to prime factor a number like 27?" After the students are sure what is meant by prime factoring, then the next question the teacher may ask is "How would we prime factor 27?" When it is well established that to prime factor a number we need to use division and the divisors must all be primes, then the class should be given some exercises to work where they have to prime factor given numbers by hand. This will help them realize what processes the computer must use.

The second class period the actual program may be written. Again it is suggested that the teacher ask questions to help the students get the program started. To get the student response that first it is necessary to READ the number that is to be prime factored into the machine, the teacher may simply ask, "Considering the exercises that you did yesterday and trying to relate them to what the computer will do, what must our first step be?" Also after working some exercises the students will be aware that to prime factor a number it is necessary to use division. The teacher's next question could be: "Now that we have N in the machine what must we do to it?" After they have responded that division is the operation needed, ask the students, "What kind of numbers should we use as divisors?" Then ask, "How would we get the prime divisors in the computer?" To actually develop:

```
40 FOR K= 2 TO N-1       NOTE:  Refer to program on following page
       .
       .
       .
100 NEXT K
```

To introduce the primes into the computer will probably require a suggestion from the teacher. However, the teacher is cautioned here to present this idea only as a suggestion when the students have met a dilemma and can't produce an idea that will work. Then let them discuss your suggestion to decide if they think it will work.

From their knowledge of factorization the students will know that after we divide N we are interested in the quotient and that is the next number to be divided by a prime. Therefore the teacher could ask, "Now that we have divided N by one prime how should we proceed now to get the rest of the prime factors?" Statement 50 checks to see if the quotient found by dividing

N by a prime number K is an integer.

```
50 IF N<>INT(N/K)*K THEN 100
```

Here again the teacher will probably have to aid the class discussion with a suggestion. Statement 50 is an IF...THEN command. IF the quotient is not an integer which makes the statement true then the computer goes to statement 100 which increments K and the new K is tested to see if it can by a prime factor.

If the quotient is an integer then K divided N evenly and K is a factor of N. Therefore we want to print K. Next replace N with the quotient N/K so we can then check to see what prime will divide it.

It may happen that the N that is put into the computer is a prime number. To check for this, C=0 is put in as a counter. Each time a factor is printed, C is incremented by one. After each possible K is checked then C is compared to zero. If it is not equal to zero then N has some factors and is not prime. IF C=0 then N is prime so the computer prints "IS PRIME" next to N.

Complete program and flow chart (Figure 8) are on the following pages. It is also suggested that the teacher take a number that the class wants prime factored and after the program is finished go through the program using that number so the students may see exactly how the computer produces the factors. (An example follows using the Number 27).

Also it may be of benefit to have the students look for and program another approach to the same problem.

Here are some questions that may be used for a quiz after the class has completed the program.

1) Why did we test primes to N-1 as factors of N and not to N or beyond?

2) What would happen if we would put 1 in for N?

3) When we divided N by a prime number K we got a quotient. Why was it necessary to use the INT command to see if that quotient was an integer?

4) After N is replaced by the quotient N/K, we check N/K without incrementing K. Why? Can you give an example?

EXAMPLE ONE:

```
27 ÷ 2 = 13.5
INT (13.5) = 13
27 ≠ 13 * 2
THEREFORE TAKE THE NEXT K
27 ÷ 3 = 9
INT(9) = 9
27 = 9*3
THEREFORE PRINT 3
LET N = 9
9 ÷ 3 = 3
INT (3) = 3
9 = 3 * 3
THEREFORE PRINT 3
LET N = 3
3 ÷ 3 = 1
INT(1) = 1
3 + 3X1
THEREFORE PRINT 1
LET N = 1
1 ÷ 3 = .3333̄
INT (.3333̄) = 0
1 ≠ 0.1
THEREFORE TAKE THE NEXT K
1 ÷ 4 = .25
INT (.25) = 0
1 ≠ 0.1
THEREFORE TAKE THE NEXT K
.
.
.
.
THEREFORE TAKE NEXT K
1 ÷ 26 = .038
INT(.038) = 0
1 ≠ 0.1
C = 3
THEREFORE END

PRINTED
27   3   3   3
```

```
CU*DENVER**1
 SORRY HAD TØ RELØAD WITH SUNDAYS FILE
ØN AT  12:58  PX TUE 08/22/67   TTY 3
USER NUMBER--P61000
SYSTEM--BASIC
NEW ØR ØLD--NEW
NEW PRØBLEM NAME--PRIME
READY.
IP←APE
READY.
10 READ N
20 PRINT N;
30 LET C = 0
40 FØR K = 2 TØ N-1
50 IF N <> INT(N/K)*K THEN 100
60 PRINT K;
70 LET C = C+1
80 LET N = N/K
90 GØ TØ 50
100 NEXT K
110 IF C<>0 THEN 130
120 PRINT "IS PRIME"
130 PRINT
140 GØ TØ 10
150 DATA 45,89,23,48,120,10,12045
160 END
KEY
READY.

RUN
```

```
PRIME    13:00   PX TUE 08/22/67

45     3     3     5
89    IS PRIME
23    IS PRIME
48     2     2     2     2     3
120    2     2     2     3     5
10     2     5
12045     3     5     11    73

ØUT ØF DATA IN  10

TIME:  10 SECS.


BYE

*** ØFF AT 13:01   PX TUE 08/22/67.
```

FIGURE 1

A	B	Q
5	2	2.5
7	1	7
7	3	2.33333

FIGURE 2

A	B	Q	R
10	3	3	1
7	2	3	1
8	5	1	3

FIGURE 3

A	B	Q	R
12	1	12	0
12	4	3	0
12	8	1	4
12	13	0	12

FIGURE 4

A	B	Q	R
12	1	12	0
12	2	6	0
12	3	4	0
12	4	3	0
12	6	2	0
12	12	1	0

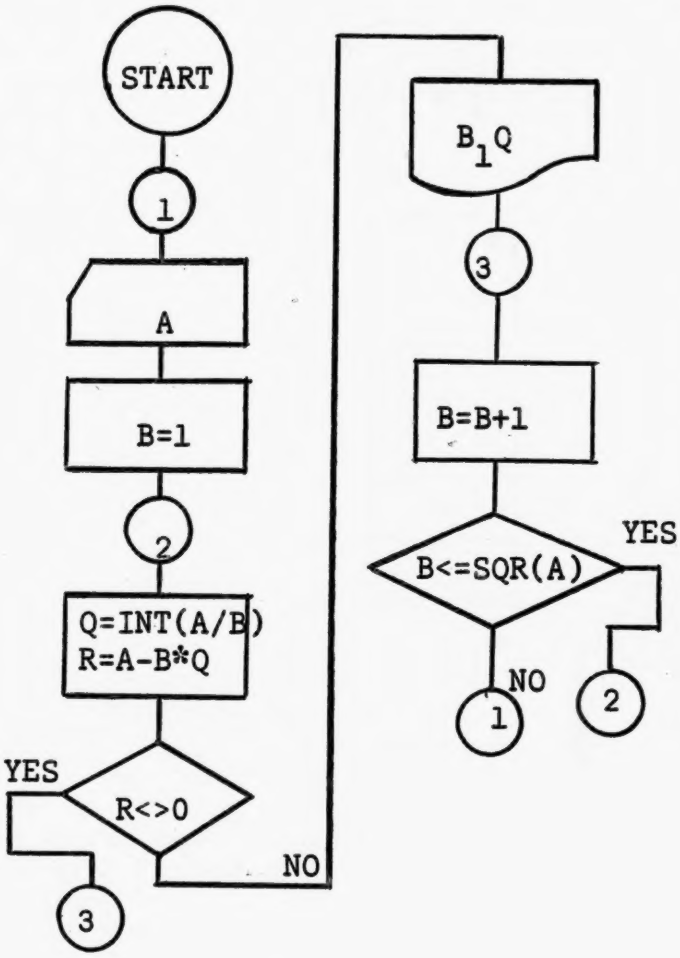

START
1
A
B=1
2
Q=INT(A/B)
R=A-B*Q
R<>0
YES
3
NO
B,Q
3
B=B+1
B<=SQR(A)
YES
NO
1
2

FIGURE 5

FIGURE 6

A	B	Q	R
12	16	0	12
16	12	1	4
12	4	3	0

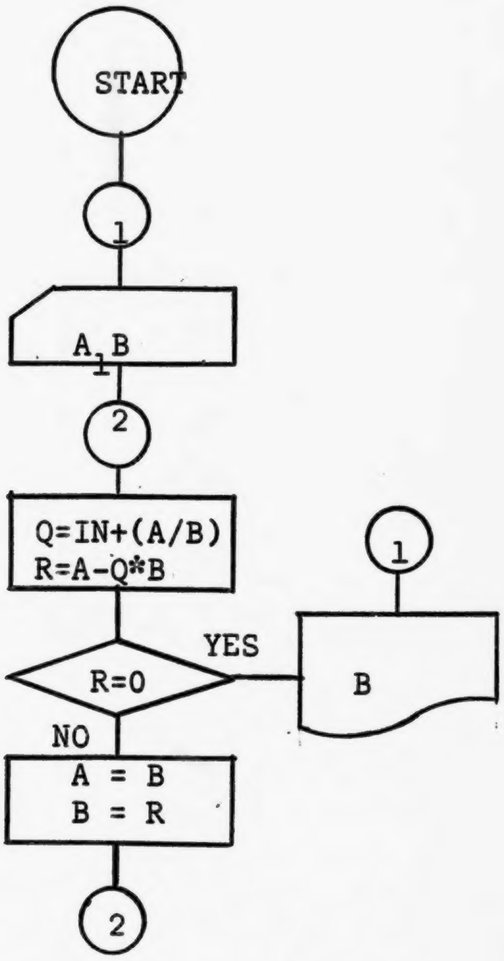
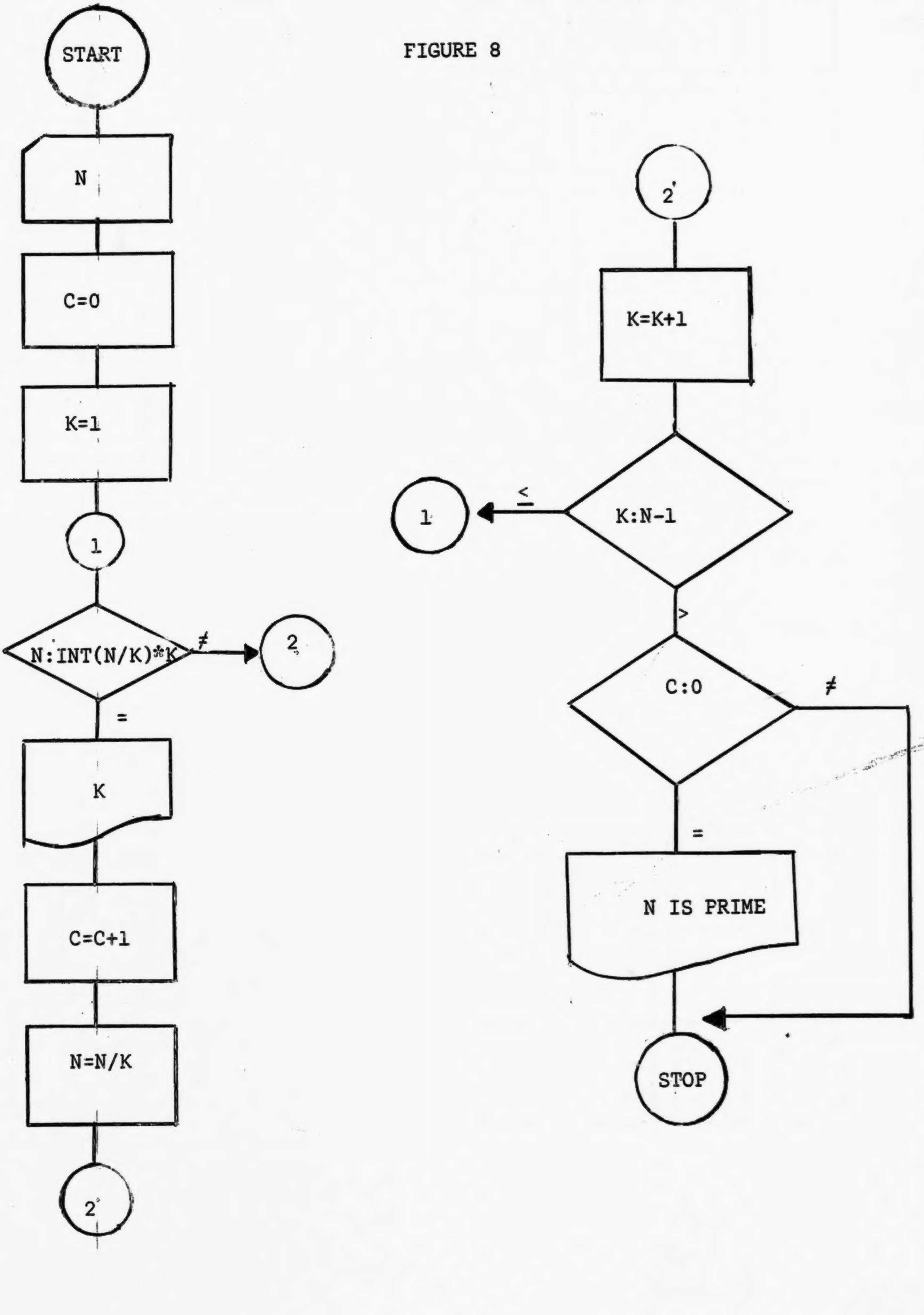

START
1
A,B
2
Q=IN+(A/B)
R=A-Q*B
R=0
YES
B
1
NO
A = B
B = R
2

FIGURE 7

FIGURE 8


START
N
C=0
K=1
1
N:INT(N/K)*K
≠
2
=
K
C=C+1
N=N/K
2
2
K=K+1
K:N-1
≤
1
>
C:0
≠
=
N IS PRIME
STOP

DATA EVALUATION FOR THE "CORRECT ANSWER" FOR JUNIOR HIGH SCHOOL STUDENTS

by Margaret Ann Grill

Purpose of Paper

The purpose of this paper is to present a procedure for evaluation of student data obtained from experiments in ninth grade physical science. Since tedious numerical methods are often used in such an evaluation, much time can be saved by using the computer; and this will be illustrated.

Introduction

A junior high student cannot expect nor be expected to get accepted literature values (those found in standard references such as the "Handbook of Chemistry and Physics") from his experimental endeavors or those values which are calculated from a given mathematical statement. If his results are compared to these values, the result is nothing more than the calculation of his percent error. However, he is eager to know if he has "the correct answer." An accepted "correct answer" for a junior high student should be based upon his:

1. Limited number of trials.
2. Limited technique.
3. Limited experimental procedure.
4. Limited equipment and materials.
5. Limited mathematical ability.

To determine the "correct answer" with the given limitations, the student will gain valuable experience if he participates in the evaluation of the data which will result in the "correct answer." If his results are outside of the accepted bounds, perhaps he will be motivated to analyze his work and find out how, where and why he "goofed." He should be encouraged to repeat parts of the experiment to verify his analysis.

Procedure

To illustrate evaluation of student data, it is assumed that all the students have completed the experiment and that the teacher has a record of all the data. Transparencies are prepared prior to the class discussion. Each set of data to be evaluated will require one transparency. To facilitate the discussion, the data should be recorded on the upper portion of the transparency. This could be done by a student assistant. See Figure 1 for an example. Results ("answers") from an unsophisticated study of the inverse square law in ninth grade physical science will be used as data. To start the discussion, ask the students to search the data and find the smallest number. Use this number as a label for the extreme left column of the graph on the transparency. Continue with the next

TRANSPARENCY EXAMPLE

Data:

4.6, 5.0, 4.8, 5.0, 4.9, 5.1, 5.1, 4.9, 5.0, 5.2,
5.0, 5.0, 4.7, 5.2, 5.1, 4.7, 4.8, 5.0, 5.2, 5.1,
4.9, 6.0, 5.5, 4.5, 5.0, 4.9, 5.6, 4.8, 5.5, 4.8

(unit in centimeters)

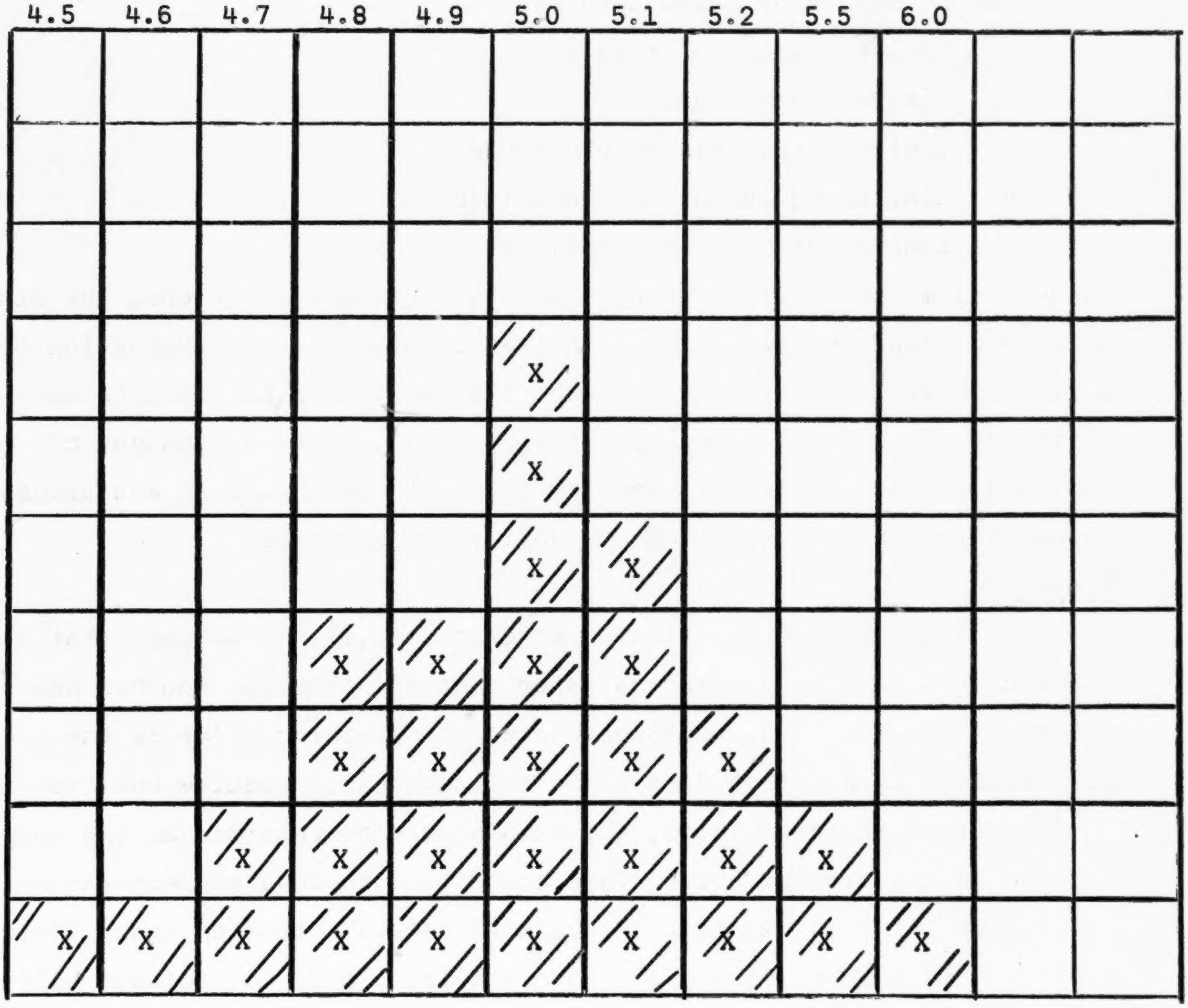


Figure 1

largest number, etc., until the columns of the graph are labeled in order from the smallest number at the left to the largest number on the right. See Figure 1 for an example. To enter the data on the graph, have a student read the first number. Mark an "x" to represent this number in the bottom space of the appropriate column. If this number occurs again, it will occupy the next space up. After all numbers are represented on the graph, the occupied spaces could be filled in with a colored pen. The result would be a bar graph or a hologram. At this time, ask the students to recall the units for the numbers. For this example, the unit is the centimeter. Have them consider how much one number varies from the next, which columns have the most numbers, which have the least, etc. From this, they should arrive at some agreement on which numbers occur most frequently. They will probably select the numbers from 4.7 to 5.2. A discussion should follow deciding whether these "answers" are reasonable for the procedure of the experiment. Probably these are the "correct answers" for this one set of data.

Ask the students how they would find the average (mean) of the numbers. Introduce the concept of deviation from the mean. Guide the discussion in such a way that they will understand the meaning of standard deviation. After they have some idea of this, present the mathematical expression for the calculation of the standard deviation. Ask how many of them would like to make these calculations for the nine sets of data from this experiment. Have them estimate how long these calculations would take. If a computer is available, introduce its use at this time for these repeated calculations. Put the tape (which has been prepared prior to the discussion) of the program into the computer, and run it. Note the calculation time for the computer, and compare this with the previous estimation by human calculations. See Figure 2 and following pages for this program, etc.

Explain that the computer can do only that which it is instructed to do in its own language. Put a transparency of the flow chart for the program on the projector, and briefly explain how the program is logically executed.

Use the standard deviation obtained from the computer calculation to determine the acceptable range for the "correct answers." For such small numbers, it would be best to allow only one standard deviation. The "correct answers" for this set of data would be from 4.7 to 5.3. Have the students compare these with the results obtained from the hologram. It would be well to have a short discussion of rounding off numbers and the possible errors involved in general.

FLOW CHART FOR SORTING A LIST OF DATA, CALCULATION OF DEVIATIONS, MEAN AND STANDARD DEVIATION

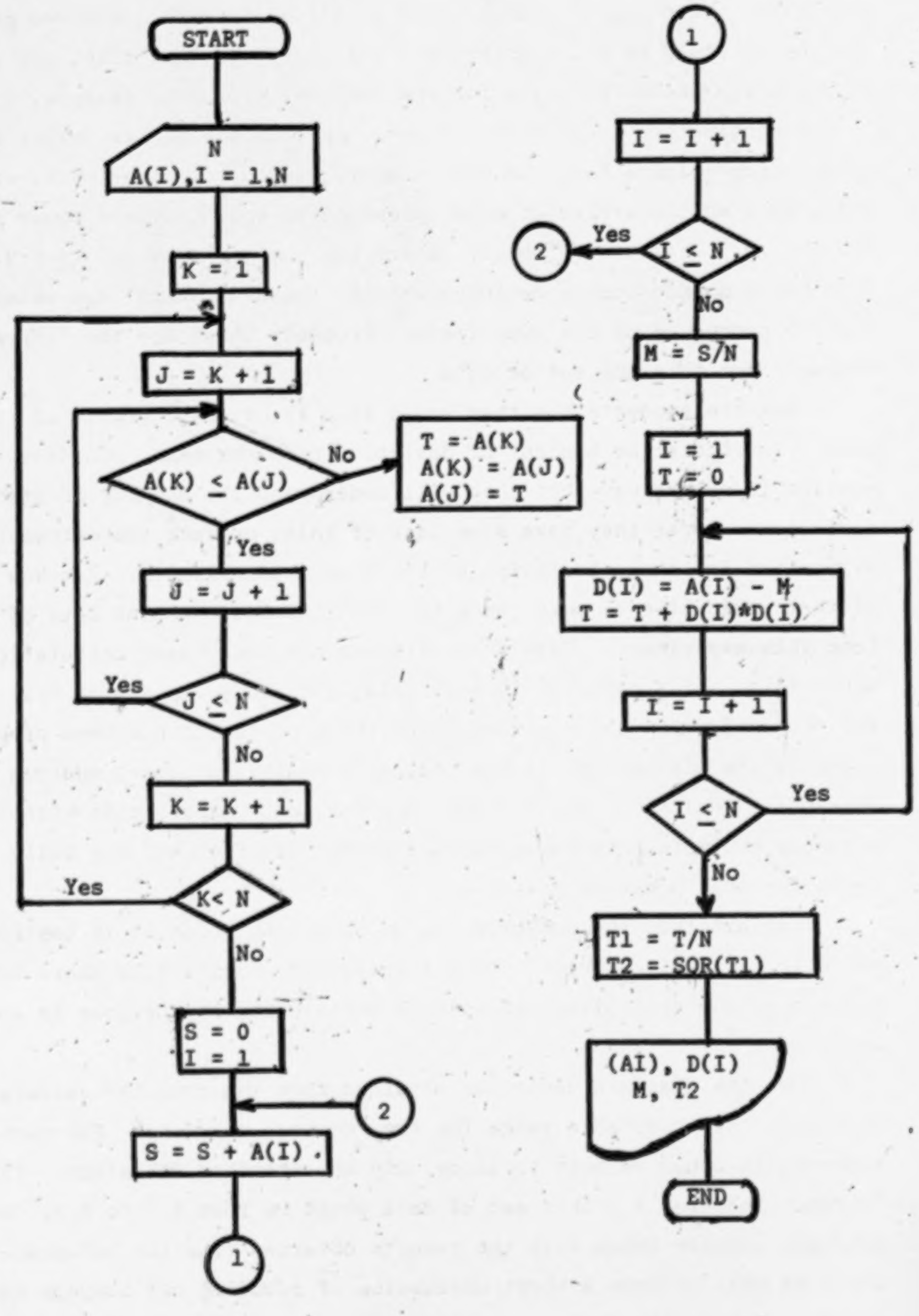

START
N
A(I),I = 1,N
K = 1
J = K + 1
A(K) ≤ A(J)
No
T = A(K)
A(K) = A(J)
A(J) = T
Yes
J = J + 1
Yes
J ≤ N
No
K = K + 1
Yes
K< N
No
S = 0
I = 1
2
S = S + A(I)
1
1
I = I + 1
Yes
2
I ≤ N
No
M = S/N
I = 1
T = 0
D(I) = A(I) - M
T = T + D(I)*D(I)
I = I + 1
I ≤ N
Yes
No
T1 = T/N
T2 = SQR(T1)
(AI), D(I)
M, T2
END

Figure 2

COMPUTER PROGRAM
FOR SORTING A LIST OF DATA, CALCULATION OF MEAN, DEVIATIONS AND STANDARD DEVIATION

```
GRILL1     15:03     PX THU 07/13/67

10 DIM N(30), A(30), D(30)
20 PRINT "N=", "D="
30 READ N
40 FØR I = 1 TØ N
50 READ A(I)
60 NEXT I
70 LET K = 1
80 LET J = K + 1
90 IF A(K) < = A(J) THEN 130
100 LET T = A(K)
110 LET A(K) = A(J)
120 LET A(J) = T
130 LET J = J + 1
140 IF J <= N THEN 90
150 LET K = K + 1
160 IF K < N THEN 80
200 LET S = 0
210 FØR I = 1 TØ N
220 LET S = S + A(I)
230 NEXT I
240 LET M = S/N
250 LET T = 0
260 FØR I = 1 TØ N
270 LET D(I) = A(I) - M
280 LET T = T + D(I)*D(I)
290 NEXT I
300 LET T1 = T/N
310 LET T2 = SQR (T1)
320 FØR I = 1 TØ N
330 PRINT A(I), D(I)
340 NEXT I
350 PRINT "M=" M, "S.D.=" T2
355 PRINT
360 GØ TØ 20
660 END
```

In general, the listed statements are instructing the computer to do the following:

10-60	Put data into memory.
70-160	Sort the random numbers.
200-230	Calculate the sum.
240	Calculate the mean.
250-270	Calculate the deviation of each number from the mean.
280-310	Calculate the standard deviation.
320-350	Print sorted numbers with the deviation of each, mean and standard deviation.
355	Allow a blank space before printing again.
360	Pick up next set of data.

Data statements would be numbered from 361 to 659.

660	Stop computing.

The example below illustrates the data instruction using the first set of data.

```
365 DATA 30
370 DATA 4.6,5.0,4.8,5.0,4.9,5.1,5.1,4.9, 5.0, 5.2, 5.0, 5.0,
380 DATA 4.7, 5.2,5.1, 4.7, 4.8, 5.0, 5.2, 5.1, 4.9, 6.0, 5.5,
390 DATA 4.5, 5.0, 4.9, 5.1, 4.8, 5.5, 4.8
```

COMPUTER OUTPUT
FOR THE NINE SETS OF DATA

```
GRILL1      11:15     PX THU 07/13/67

N=            D=                                        N=           D=
 4.5          -.513333                                   6.5         -.673333
 4.6          -.413333                                   6.8         -.373333
 4.7          -.313333                                   6.8         -.373333
 4.7          -.313333                                   6.9         -.273333
 4.8          -.213333                                   6.9         -.273333
 4.8          -.213333                                   6.9         -.273333
 4.8          -.213333                                   6.9         -.273333
 4.8          -.213333                                   7           -.173333
 4.9          -.113333                                   7           -.173333
 4.9          -.113333                                   7           -.173333
 4.9          -.113333                                   7           -.173333
 4.9          -.113333                                   7           -.173333
 5            -1.33333 E-2                               7.1         -7.33333 E-2
 5            -1.33333 E-2                               7.1         -7.33333 E-2
 5            -1.33333 E-2                               7.1         -7.33333 E-2
 5            -1.33333 E-2                               7.1         -7.33333 E-2
 5            -1.33333 E-2                               7.1         -7.33333 E-2
 5            -1.33333 E-2                               7.2          2.66667 E-2
 5            -1.33333 E-2                               7.2          2.66667 E-2
 5.1           8.66667 E-2                               7.2          2.66667 E-2
 5.1           8.66667 E-2                               7.2          2.66667 E-2
 5.1           8.66667 E-2                               7.4          .226667
 5.1           8.66667 E-2                               7.4          .226667
 5.1           8.66667 E-2                               7.4          .226667
 5.2           .186667                                   7.5          .326667
 5.2           .186667                                   7.5          .326667
 5.2           .186667                                   7.5          .326667
 5.5           .486667                                   7.5          .326667
 5.5           .486667                                   8            .826667
 6             .986667                                   8            .826667
M= 5.01333    S.D.= .286046                             M= 7.17333   S.D.= .324482

N=            D=
 8            -.84           9          .16
 8.2          -.64           9          .16
 8.3          -.54           9          .16
 8.4          -.44           9          .16
 8.4          -.44           9          .16
 8.4          -.44           9.2        .36
 8.5          -.34           9.2        .36
 8.5          -.34           10         1.16
 8.5          -.34           10         1.16
 8.6          -.24           10         1.16
 8.6          -.24          M= 8.84    S.D.= .480694
 8.7          -.14
 8.7          -.14
 8.8          -3.99999 E-2
 8.8          -3.99999 E-2
 8.8          -3.99999 E-2
 8.9          .06
 8.9          .06
 8.9          .06
 8.9          .06
```

```
N=           D=
 9.5         -.876667
 9.7         -.676667
 9.8         -.576667
 9.8         -.576667
 9.9         -.476667
 9.9         -.476667
 10          -.376667
 10          -.376667
 10          -.376667
 10          -.376667
 10          -.376667
 10          -.376667
 10.1        -.276667
 10.1        -.276667
 10.1        -.276667
 10.1        -.276667
 10.2        -.176667
 10.2        -.176667
 10.3        -7.66666 E-2
 10.3        -7.66666 E-2
 10.3        -7.66666 E-2
 10.3        -7.66666 E-2
 10.4         2.33334 E-2
 10.5         .123333
 10.5         .123333
 11.2         .823333
 11.5         1.12333
 12           1.62333
 12           1.62333
 12.6         2.22333
M= 10.3767   S.D.= .725113
```

```
N=          D=
 10.8       -.82
 11         -.62
 11         -.62
 11         -.62
 11.1       -.52
 11.2       -.42
 11.2       -.42
 11.2       -.42
 11.2       -.42
 11.2       -.42
 11.2       -.42
 11.2       -.42
 11.3       -.32
 11.3       -.32
 11.3       -.32
 11.3       -.32
 11.3       -.32
 11.4       -.22
 11.4       -.22
 11.5       -.12
 11.5       -.12
 11.7        8.00001 E-2
 12          .38
 12          .38
 12.2        .58
 12.5        .88
 12.7        1.08
 13          1.38
 13.2        1.58
 13.7        2.08
M= 11.62    S.D.= .713396
```

```
N=           D=
 12          -.813333        14          1.18667
 12          -.813333        14          1.18667
 12          -.813333        14.8        1.98667
 12.1        -.713333        14.9        2.08667
 12.1        -.713333        15          2.18667
 12.1        -.713333       M= 12.8133  S.D.= .882698
 12.1        -.713333
 12.2        -.613333
 12.2        -.613333
 12.2        -.613333
 12.3        -.513333
 12.3        -.513333
 12.3        -.513333
 12.3        -.513333
 12.4        -.413333
 12.6        -.213333
 12.6        -.213333
 12.6        -.213333
 12.7        -.113333
 12.7        -.113333
 12.9         8.66668 E-2
 13.1         .286667
 13.1         .286667
 13.2         .386667
 13.6         .786667
```

```
N=          D=                                        N=          D=
 12.9       -1.08333                                   13.8       -1.03333
 13         -.983333                                   14         -.833333
 13.1       -.883333                                   14         -.833333
 13.1       -.883333                                   14.1       -.733333
 13.1       -.883333                                   14.1       -.733333
 13.2       -.783333                                   14.1       -.733333
 13.2       -.783333                                   14.1       -.733333
 13.2       -.783333                                   14.1       -.733333
 13.3       -.683333                                   14.2       -.633333
 13.4       -.583333                                   14.3       -.533333
 13.5       -.483333                                   14.3       -.533333
 13.5       -.483333                                   14.3       -.533333
 13.5       -.483333                                   14.3       -.533333
 13.5       -.483333                                   14.5       -.333333
 13.6       -.383333                                   14.5       -.333333
 13.8       -.183333                                   14.5       -.333333
 13.8       -.183333                                   14.5       -.333333
 13.8       -.183333                                   14.6       -.233333
 13.8       -.183333                                   14.6       -.233333
 14          1.66667 E-2                               14.7       -.133333
 14          1.66667 E-2                               14.8       -3.33333 E-2
 14.1        .116667                                   15          .166667
 14.8        .816667                                   15.5        .666667
 15          1.01667                                   15.8        .966667
 15.1        1.11667                                   16          1.16667
 15.2        1.21667                                   16          1.16667
 15.2        1.21667                                   16.1        1.26667
 15.7        1.71667                                   16.7        1.86667
 16          2.01667                                   16.7        1.86667
 16.1        2.11667                                   16.8        1.96667
M= 13.9833  S.D.= .926673                             M= 14.8333  S.D.= .891191

N=          D=
 14.7       -.966667       16.9        1.23333
 14.8       -.866667       17          1.33333
 14.9       -.766667       17          1.33333
 14.9       -.766667       17.4        1.73333
 15         -.666667       17.5        1.83333
 15         -.666667       17.5        1.83333
 15         -.666667      M= 15.6667  S.D.= .888194
 15         -.666667
 15         -.666667
 15.1       -.566667
 15.1       -.566667
 15.1       -.566667
 15.1       -.566667
 15.2       -.466667
 15.2       -.466667
 15.2       -.466667
 15.3       -.366667
 15.4       -.266667
 15.5       -.166667
 15.6       -6.66666 E-2
 15.8        .133333
 15.8        .133333
 16.2        .533333
 16.8        1.13333
```

Continue the evaluation of each set of data. Using the sorted list of numbers from the computer calculations will shorten the time for the preparation of more holograms. Have various students read these so they will see the "answers" from the computer. It would be interesting for the students to know that the computer sorted the random set of numbers into successive order starting with the smallest. Probably after the completion of the second hologram the students will accept the results from the computer calculations for the remaining sets of data.

If a computer is available in the school, students should be given a program for their calculations if the calculations are time-consuming. The use of the computer by the student would be a great motivator.

In this experiment, two graphs are made and compared. The result shows that if the illumination is plotted versus the distance a definite curve is obtained, and that the illumination versus the inverse of the square of the distance results in a straight line. The more able student should be able to derive the equation for the straight line using geometrical concepts and algebra. See Figure 3. Another computer demonstration could be presented to do the tedious arithmetic involved in deriving this equation by the method of least squares. For data from this experiment the linear equation is "$y = 0.04x$," since -0.00058 is essentially zero. See Figure 4 and the following pages. The significance of the straight line obtained should be discussed.

Summary

A procedure involving the use of the computer to evaluate data resulting from students' experimental work has been presented.

The use of a computer to statistically evaluate data relieves the teacher of many tedious and time-consuming calculations. If this technique were available for use on data collected from experimental work by students, the teacher should be more willing to do the large number of calculations required for this evaluation. Changes, then, in experimental procedure could be suggested which would result in more meaningful data for the student. This modified procedure would enable future students to better understand primary concepts resulting from experimental activities in introductory physical science.

ILLUMINATION VERSUS DISTANCE

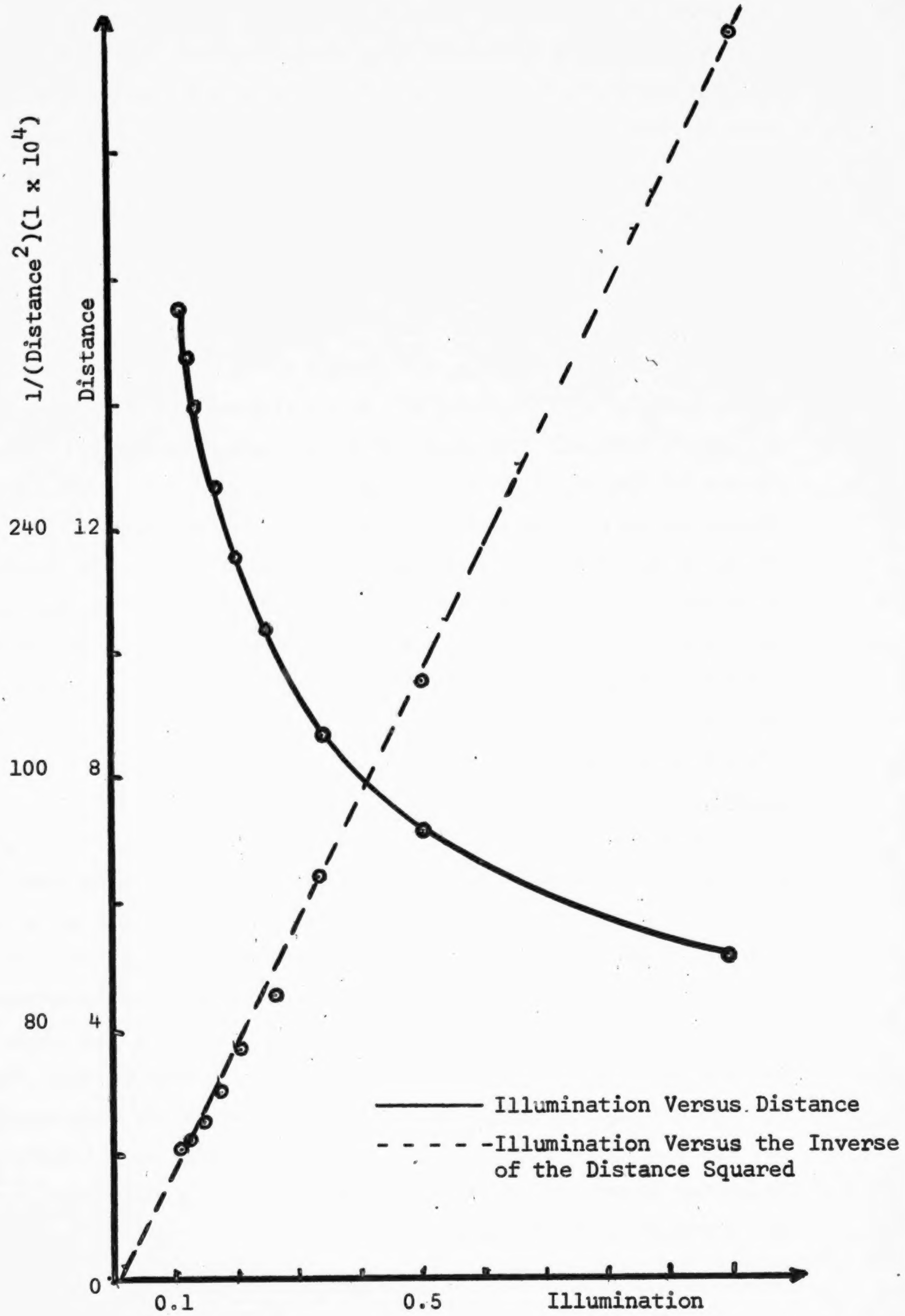


Figure 3

FLOW CHART
FOR METHOD OF LEAST SQUARES

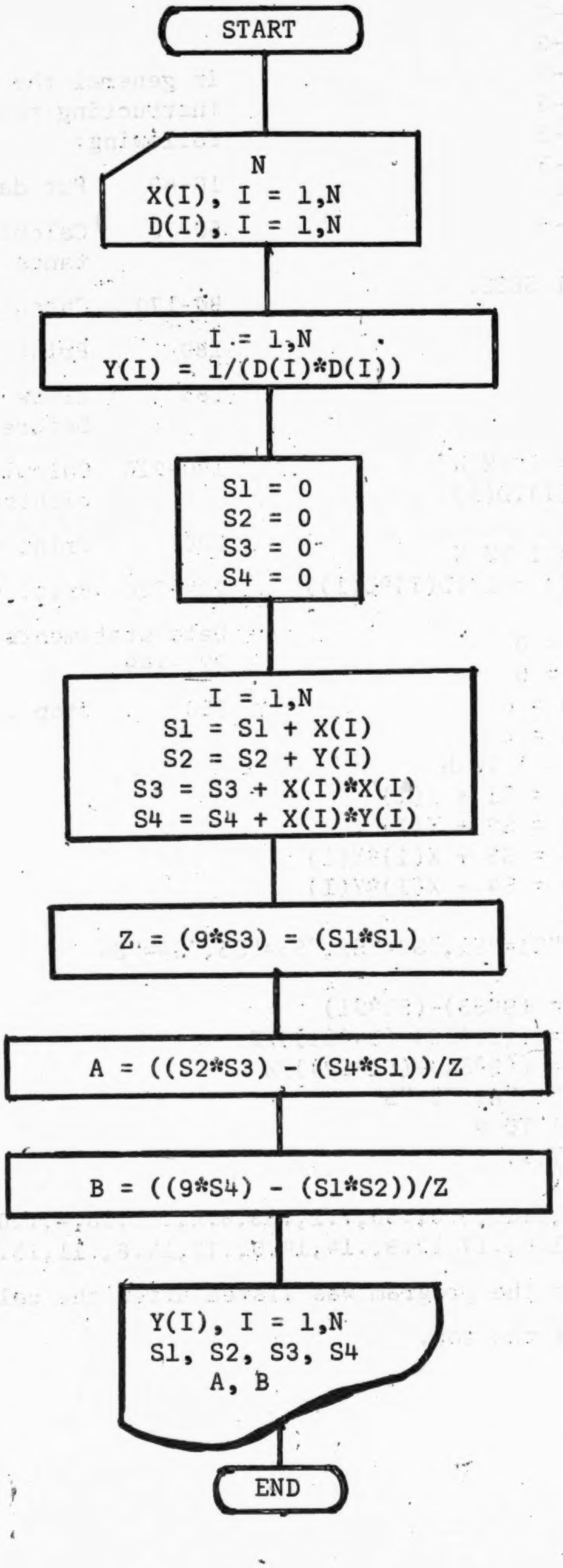

START
N
X(I), I = 1,N
D(I), I = 1,N
I = 1,N
Y(I) = 1/(D(I)*D(I))
S1 = 0
S2 = 0
S3 = 0
S4 = 0
I = 1,N
S1 = S1 + X(I)
S2 = S2 + Y(I)
S3 = S3 + X(I)*X(I)
S4 = S4 + X(I)*Y(I)
Z = (9*S3) = (S1*S1)
A = ((S2*S3) - (S4*S1))/Z
B = ((9*S4) - (S1*S2))/Z
Y(I), I = 1,N
S1, S2, S3, S4
A, B
END

Figure 4

COMPUTER PROGRAM FOR METHOD OF LEAST SQUARES

```
S1= 2.82      S2= .108708        S3= 1.5364      S4= 6.04501 E-2

A=-5.87172 E-4                   B= .040423
 .04
 1.92901 E-2
 1.29132 E-2
 9.24556 E-3
 7.43163 E-3
 6.10352 E-3
 5.10204 E-3
 4.56538 E-3
 4.05696 E-3

TIME:     1 SECS.
```

In general the listed statements are instructing the computer to do the following:

10-40	Put data into memory.
50-70	Calculate the square of the distance and then the inverse.
80-170	Calculate various sums,
180	Print the sums.
185	Allow a blank space (line) before printing again.
190-210	Calculate values for the coefficients of "x" and "y".
220	Print coefficients.
224-226	Print y values.

Data statements would be numbered from 227-249

250	Stop computing.

```
LISTNH
10 READ N
20 FØR I = 1 TØ N
30 READ X(I),D(I)
40 NEXT I
50 FØR I = 1 TØ N
60 LET Y(I) = 1/(D(I)*D(I))
70 NEXT I
80 LET S1 = 0
90 LET S2 = 0
100 LET S3 = 0
110 LET S4 = 0
120 FØR I = 1 TØ N
130 LET S1 = S1 + X(I)
140 LET S2 = S2 + Y(I)
150 LET S3 = S3 + X(I)*X(I)
160 LET S4 = S4 + X(I)*Y(I)
170 NEXT I
180 PRINT "S1="S1,"S2="S2,"S3="S3,"S4="S4
185 PRINT
190 LET Z = (9*S3)-(S1*S1)
200 LET A = ((S2*S3)-(S4*S1))/Z
210 LET B = ((9*S4)-(S1*S2))/Z
220 PRINT "A="A, "B="B
224 FØR I=1 TØ N
225 PRINT Y(I)
226 NEXT I
230 DATA 9,1.00,5.0,.50,7.2,.33,8.8,.25,10.4,.20
240 DATA 11.6,.17,12.8,.14,14.0,.12,14.8,.11,15.7
```

<u>Note</u>: Since the program was listed after the calculations, the output is at the top.

MATHEMATICAL EXPRESSIONS

For the Mean

$$M = S/N$$

where M = mean
S = sum
N = total number of values

For the Deviations

$$D = A - M$$

where D = deviation
A = value from data
M = mean

For the Standard Deviation

$$SD = \sqrt{\frac{\Sigma_{t=1}^{n} D_i^2}{n}}$$

where SD = standard deviation
Σn = sum of all values from 5 = 1 to n
t = 1
n = total number of values
D = deviation for each value

For the Method of Least Squares

$$(1) \quad AN + B\Sigma x_k = \Sigma y_k$$

$$(2) \quad A\Sigma x_k + B\Sigma x_k{}^2 = \Sigma x_k y_k$$

where N = number of values used
k = summations from k = 1 to N
x,y = values from data
A,B = constants to be solved

Let S1 = sum of x values
S2 = sum of y values
S3 = sum of x^2 values
S4 = sum of xy values

Substituting in (1) and (2) above

$$9A + S1B = S2$$
$$S1A + S3B = S4$$

Solution for A and B

$$A = \frac{(S2)(S3) - (S4)(S1)}{(9)(S3) - (S1)(S1)}$$

$$B = \frac{(9)(S4) - (S1)(S2)}{(9)(S3) - (S1)(S1)}$$

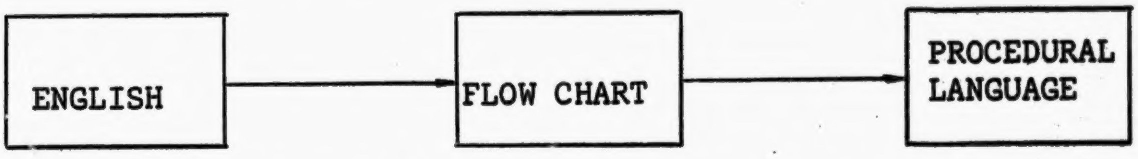
The resulting linear equation is "y = Bx + A"

INTRODUCTION TO FLOW CHARTING

By J. Groninger & T. Shoemaker

A flow chart is a pictorial diagram that can help one see his way through any logical process, sometimes not associated with a computer problem or exercise. Flow charts are commonly used in many fields of science and business. Scientists and mathematicians use flow charts to assist them in preparing problems to be solved with the aid of digital computers.

There are several translation steps necessary for preparing information for using computers. The task of using the computer to help solve problems can be separated into two steps: that of reducing the problem to a sequence of elementary steps (flow charting or preparing an algorithm); and that of converting to a procedural language. Each of these steps is a translation process.

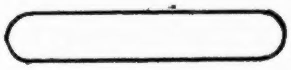


It should be remembered there is no one way of preparing a flow chart or program for solving a problem. Also, there is not a predetermined length or limit of complexity of a flow chart. The thoroughness of a translation is dependent on the desired depth of logic and difficulty of the program.

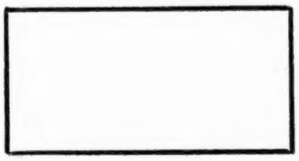
Sample Flow Charts

Several sample flow charts are shown to illustrate the intricacies of flow chart preparation and standardized symbols. Compare the general and expanded charts of one situation shown here. Both flow charts are correct.

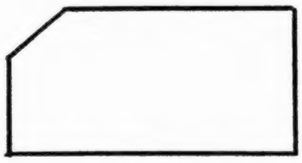
FLOW CHART SYMBOLS



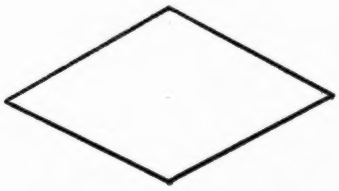
Terminal: The beginning or end of a program.



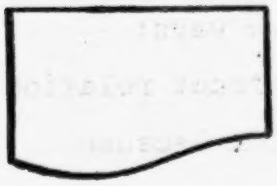
Calculation or Computing Symbol: Used to indicate a series of arithmetic operations or computer instructions.



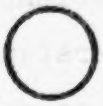
Input: Used to indicate the input of information.



Decision: A symbol used to indicate points where branching of alternate paths are possible based on variable conditions.



Output: Used to indicate an output of information.



Connector: An entry or exit to another part of a program.

Non-Math Situation

Prepare a flow chart of starting an automobile or making a local telephone call.

Mathematical Situations

Prepare a flow chart of an appropriate situation.

General Mathematics -- How to solve story problems or word problems.

Algebra I -- Finding the solution set of an equation.

Geometry -- Determining the coordinates of the midpoint of a given segment.

Algebra II -- Finding prime numbers.

Introduction to Flow Charting Teacher Information Sheet

A preparation of a flow chart probably will not make the solution of a problem easier or faster, but instead will assist the student in understanding the logical steps necessary in finding the solution. In short, a flow chart will assist the person preparing it to "see" what is going on inside the actual solution. A flow chart is the linking between a communicating everyday language and procedural language necessary for directing the necessary operations, often done on a computer. Learning to do basic flow charting will prepare you for using procedural languages such as: BASIC, COBOL, EASYCODE, etc..

Flow chart construction demands patience and the desire to prepare a detailed plan for problem solution. Students may not immediately recognize the value of a flow chart until working on a computer or computer terminal. Therefore, it is advisable for students to have a "hands-on" experience trying the logical sequential pattern of his flow chart.

The sample flow charts included are shown to assist the student in use of standardized symbols and the flow of a logically prepared diagram of solution planning. Therefore, we recommend you replace the examples provided or supplement them with situations and flow charts of your own choosing.

Teaching Techniques and Recommendations

A. Flow charts have been introduced by instructors in two ways:
   1. Through concrete problems, followed by a more abstract relation.
   2. When students are in search of a method of solution because other methods have failed or are not understood.

B. The instructor must have a working knowledge, from experience, of the assigned situation or problem. It is necessary to understand all facets of a problem solution, since there is no single correct flow chart for a problem.

C. On the situations to be completed herein the student is expected to progress at his own rate with minimal guidance from the instructor.

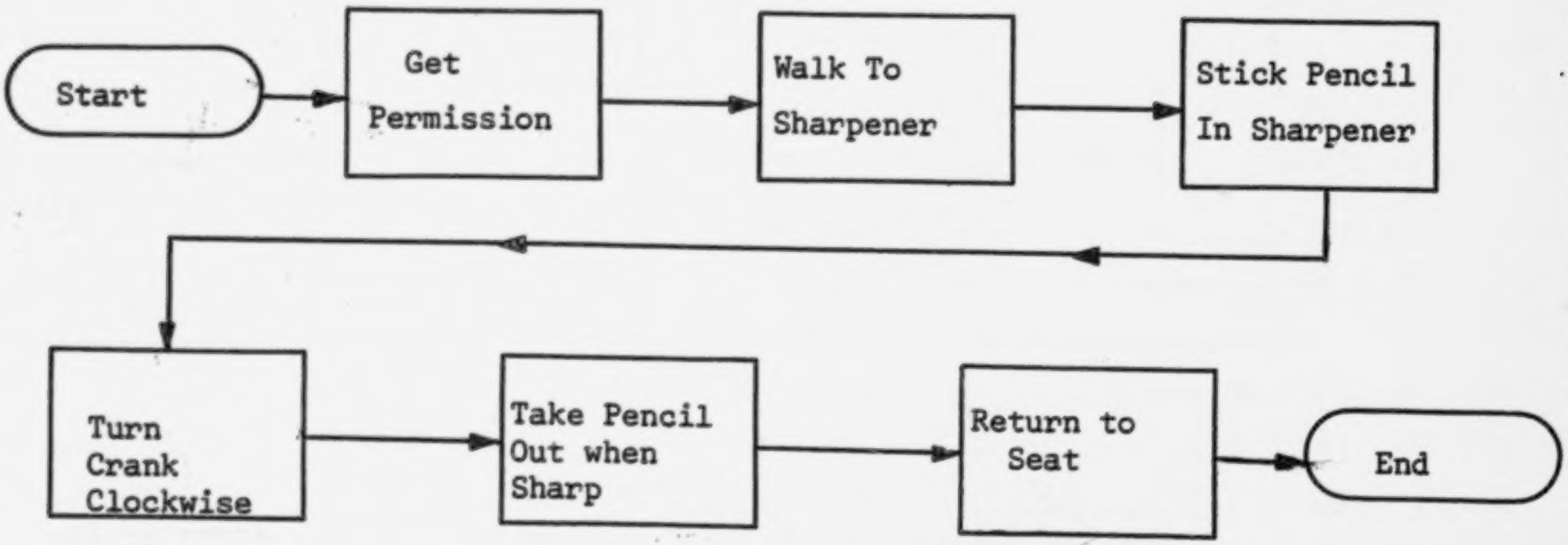
D. A flannel board with flow chart symbol cut-outs is an excellent way of demonstrating the logic pattern of a problem solution. An overhead can be used in a similar manner with plastic cutouts drawn on the projector table and projected onto a chalk board.

E. Flow chart templates can be purchased from many computer manufactures for a nominal cost.

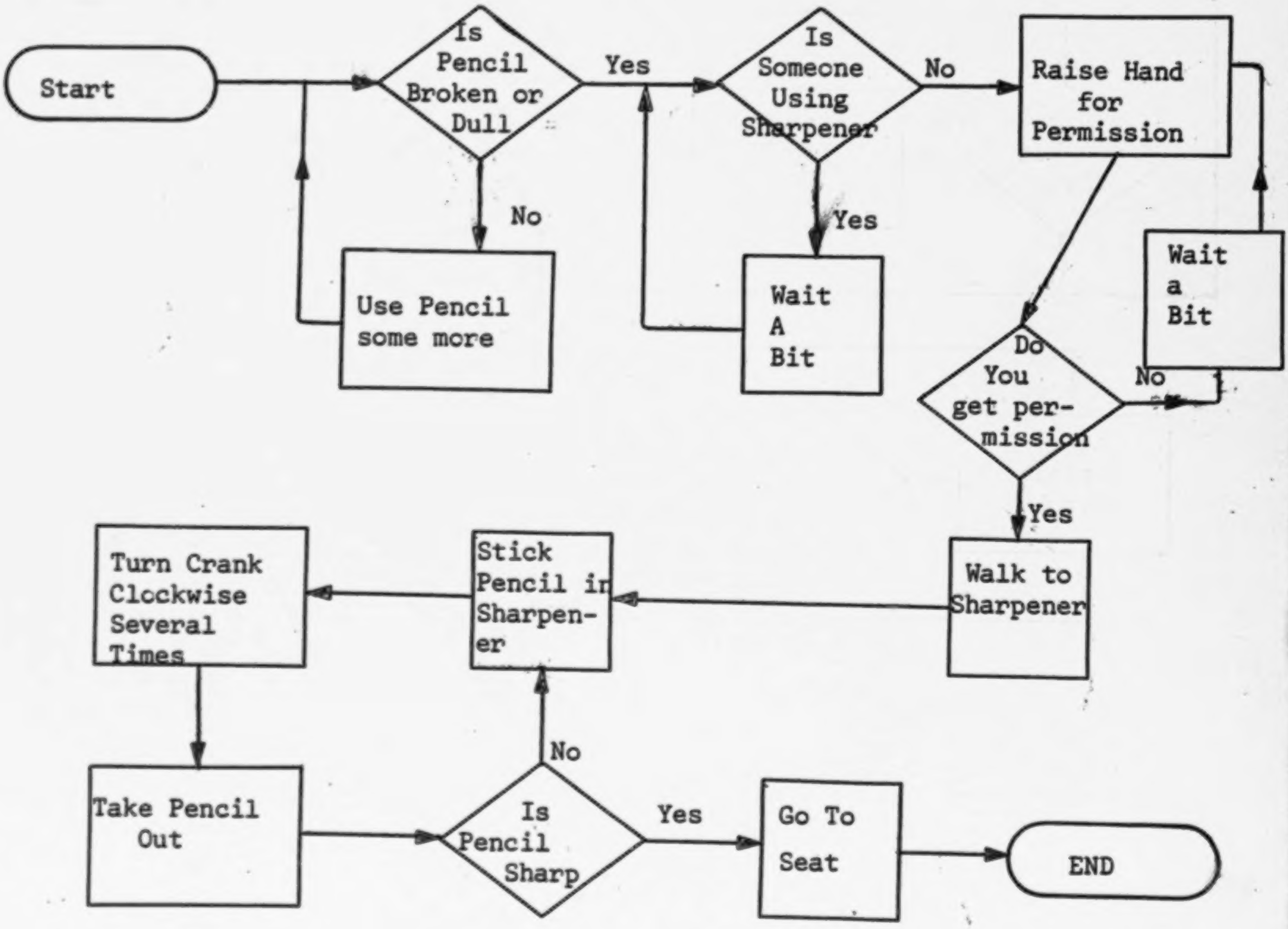
SAMPLES OF NON-MATH FLOW CHARTS

Sharpening A Pencil

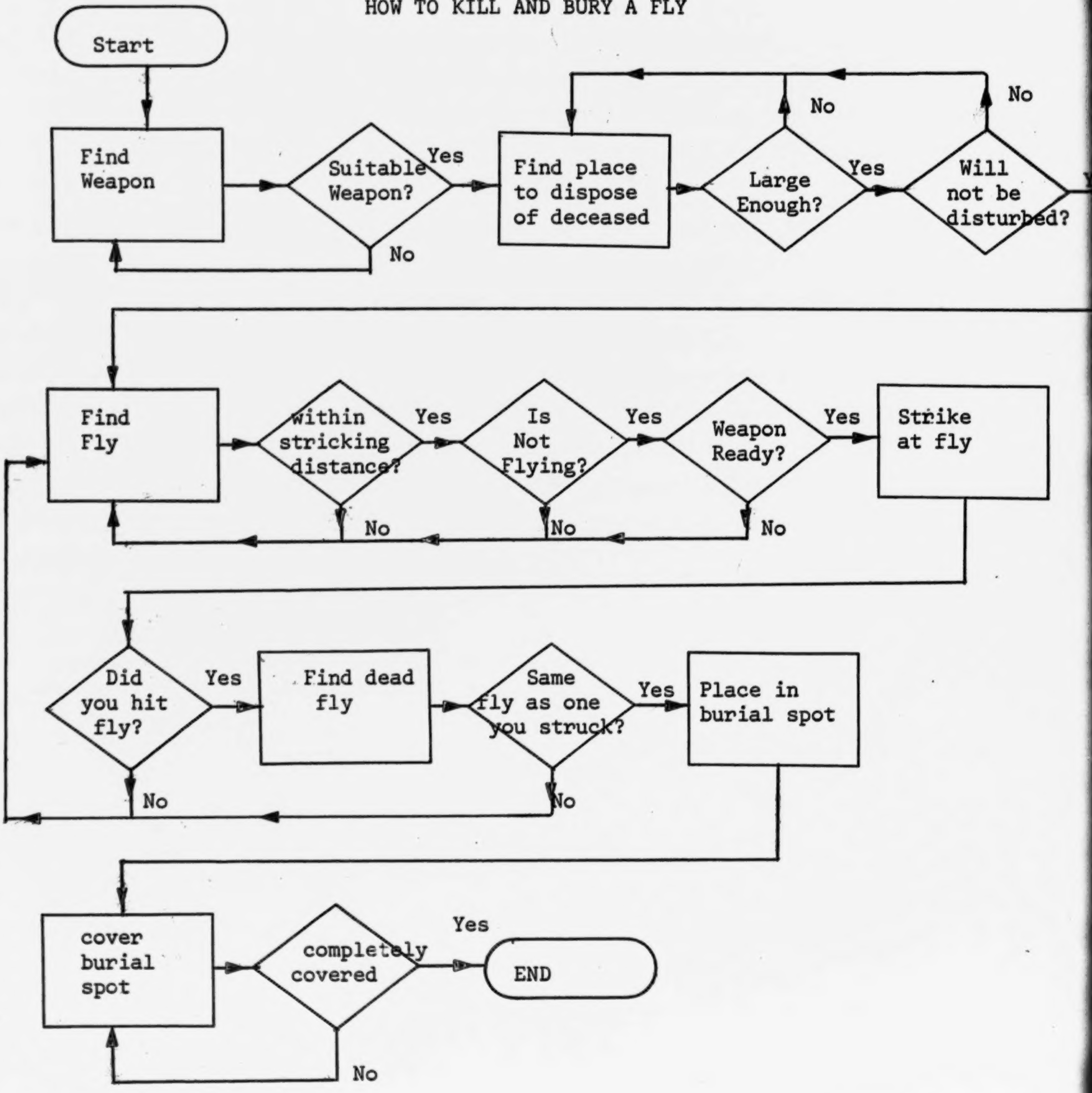
(A General Flow Chart)


Start
Get Permission
Walk To Sharpener
Stick Pencil In Sharpener
Turn Crank Clockwise
Take Pencil Out when Sharp
Return to Seat
End

(More Detailed Flow Chart)

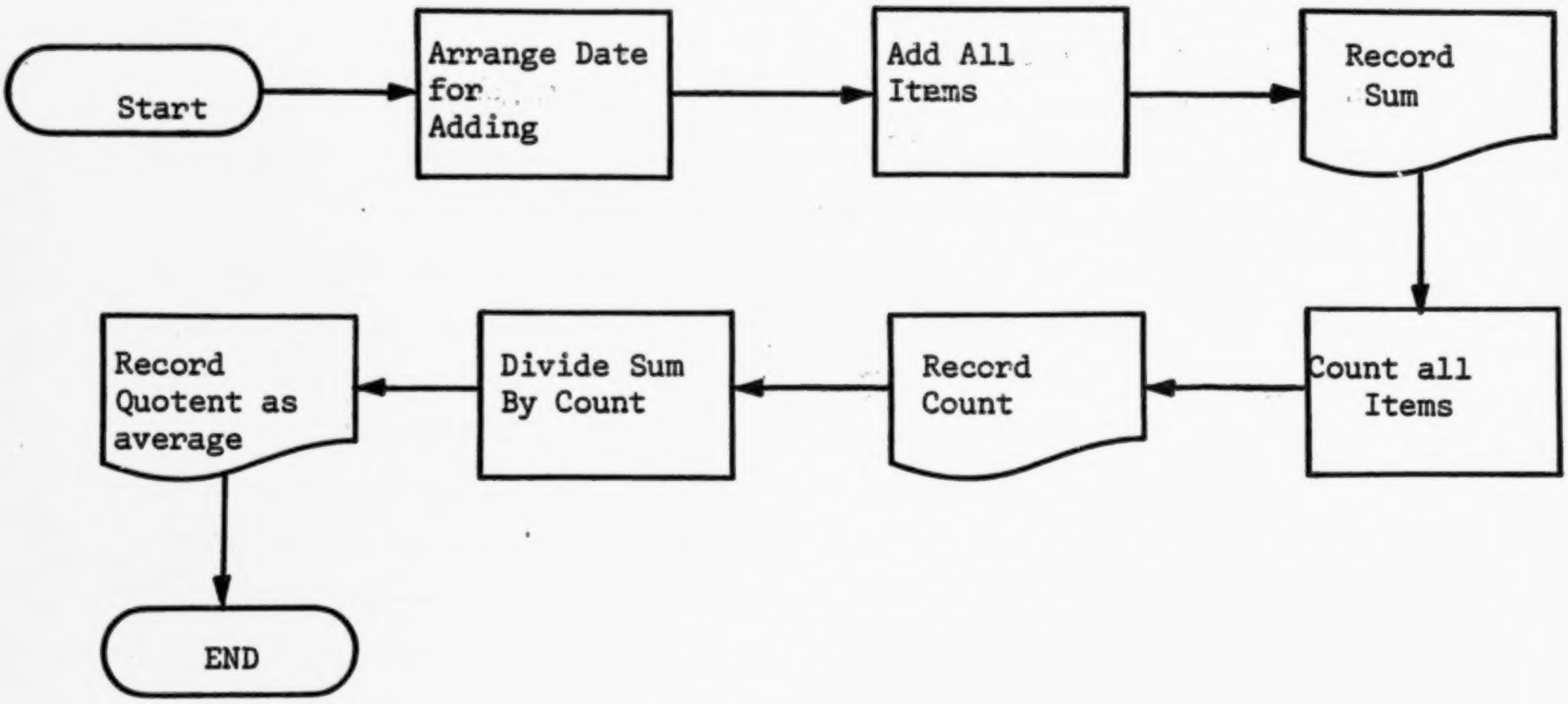

Start
Is Pencil Broken or Dull
Yes
No
Use Pencil some more
Is Someone Using Sharpener
No
Yes
Wait A Bit
Raise Hand for Permission
Do You get permission
No
Wait a Bit
Yes
Walk to Sharpener
Stick Pencil in Sharpener
Turn Crank Clockwise Several Times
Take Pencil Out
Is Pencil Sharp
No
Yes
Go To Seat
END

HOW TO KILL AND BURY A FLY

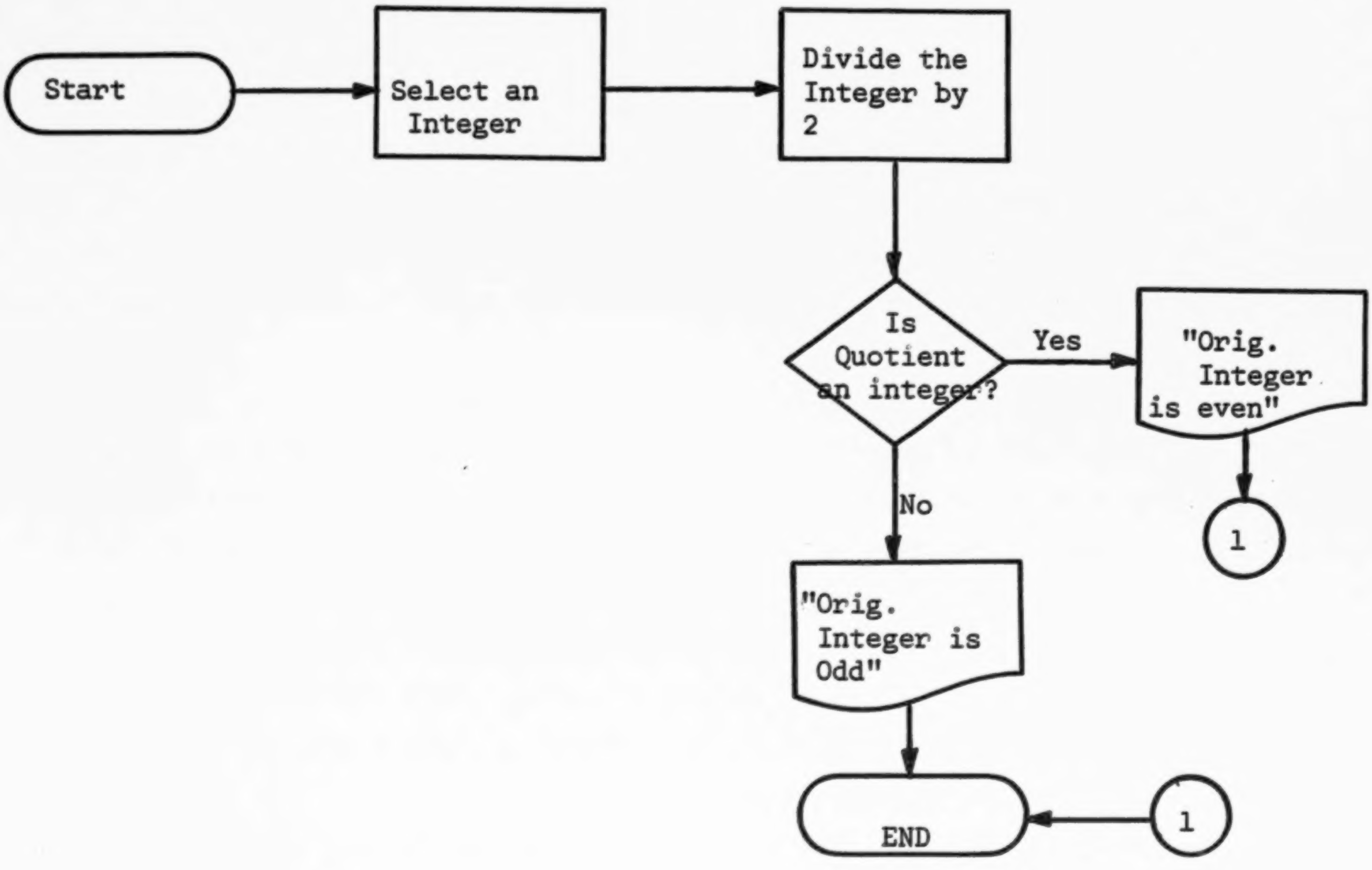

Start
Find Weapon
Suitable Weapon?
Yes
No
Find place to dispose of deceased
Large Enough?
Yes
No
Will not be disturbed?
No
Y
Find Fly
within stricking distance?
Yes
No
Is Not Flying?
Yes
No
Weapon Ready?
Yes
No
Strike at fly
Did you hit fly?
Yes
No
Find dead fly
Same fly as one you struck?
Yes
No
Place in burial spot
cover burial spot
completely covered
Yes
No
END

SAMPLES OF MATH FLOW CHART.

AVERAGING NUMBERS


Start
Arrange Date for Adding
Add All Items
Record Sum
Count all Items
Record Count
Divide Sum By Count
Record Quotent as average
END

DETERMINING WHETHER AN INTEGER IS EVEN OR ODD


Start
Select an Integer
Divide the Integer by 2
Is Quotient an integer?
Yes
"Orig. Integer is even"
1
No
"Orig. Integer is Odd"
END
1

USING THE COMPUTER FOR PERMUTATIONS AND COMBINATIONS

by Neil O. Haflich and Ellis R. Mercer

To the Teacher

The material presented in the following pages is intended to illustrate the use of the computer when working with permutations and combinations. It is written with the assumption that the student has had no previous experience with the computer. This material should be useful even if no computer is actually available to the student. This material can be used in junior high or senior high school.

The material is divided into three sections. The first section explains how a computer may be used in the calculation of factorials. It is assumed that the student is familiar with the formula ${}_nP_n = n!$. The student should have practice in calculating factorial.

The second section deals with the calculation of the numbers of permutations of n things taken r at a time. The student should have had the experience of working with the formula ${}_nP_r = \frac{n!}{(n-r)!}$.

The third section shows the calculation of the number of combinations of n things taken r at a time. The student should have had the experience of working with the formula ${}_nC_r = \frac{n!}{(n-r)!r!}$.

It is our recommendation that each of these sections involving the computer be taught following the corresponding sections in the chapter on permutations and combinations. For example, after teaching the permutations of n things taken n at a time, then teach how the computer could be used to calculate n factorial.

This method gives the student reinforcement in both the computer and the mathematical computations.

At the end of Section III it is suggested in the text that some students try to construct a flow chart and write a program for calculating ${}_nC_r$.

One possible example is shown in Figures T-1 and T-2. It should be observed that in addition to solving for ${}_nC_r$ the program also gives the value for ${}_nP_r$ which is an intermediate value in the calculation of ${}_nC_r$.

There are times when it is desirable to have data entered during the running of a program. This is particularly true when one person writes the program and enters it into the machine's memory location while other persons are to supply the data. This may be done by an "INPUT" statement, which acts as a "READ" statement but does not draw numbers from a "DATA" statement.

For example, in the program shown in Figure T-2, line 20, it reads:

Flow Chart for ${}_nP_r$ and ${}_nC_r$

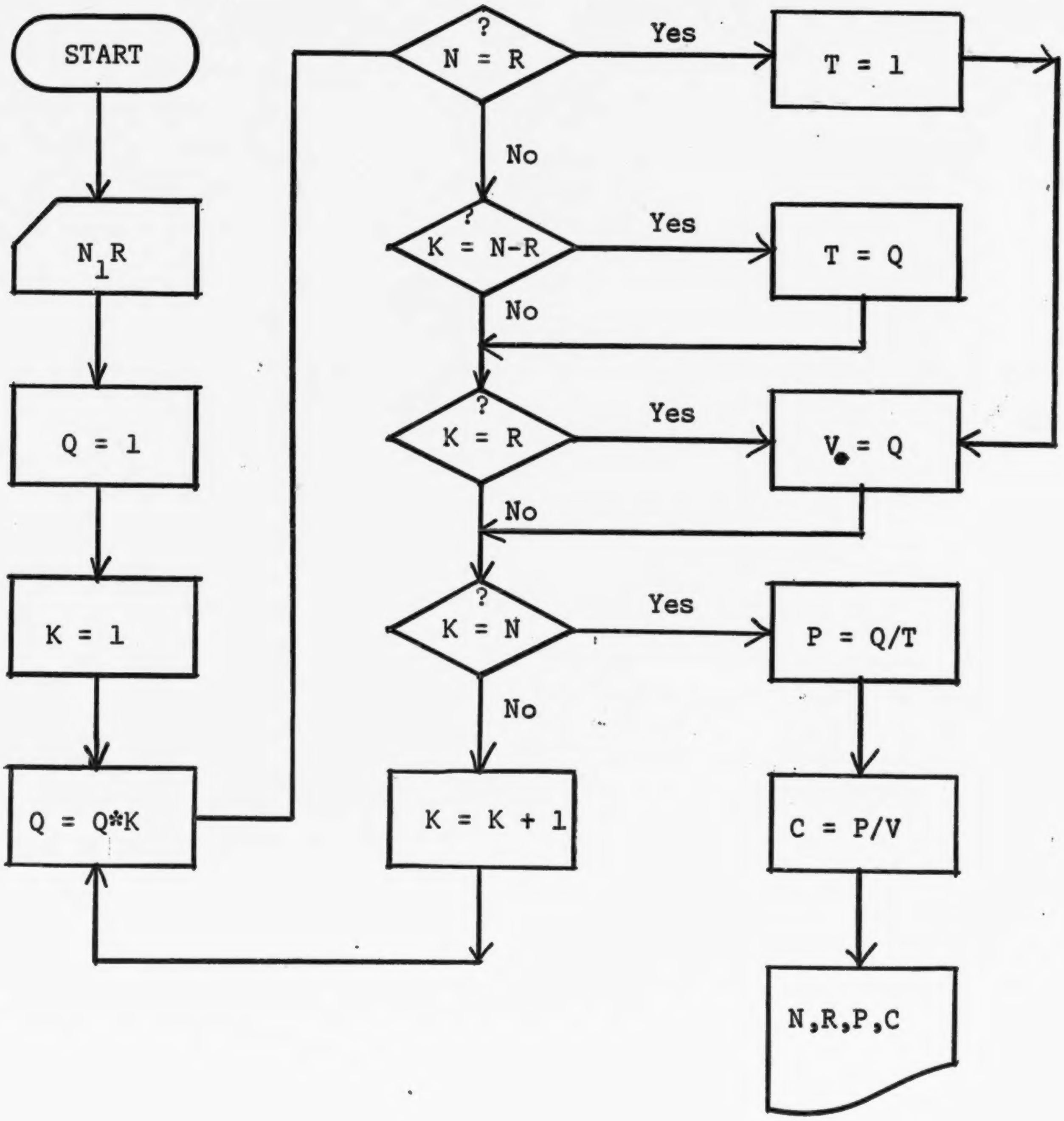

START
N, R
Q = 1
K = 1
Q = Q*K
?
N = R
Yes
T = 1
No
?
K = N-R
Yes
T = Q
No
?
K = R
Yes
V = Q
No
?
K = N
Yes
P = Q/T
No
K = K + 1
C = P/V
N,R,P,C

Figure T-1

```
20 INPUT N,R
```

This statement must come before N and R are used in the program. When the computer encounters this statement, it will type a question mark. The operator of the computer types two numbers, separated by a comma, presses the return key; and the computer goes on with the remainder of the program.

Frequently an "INPUT" statement is combined with the "PRINT" statement to make sure the user knows what the question mark is asking for. The program would then read:

```
10 PRINT "YØUR VALUES OF N AND R ARE",
20 INPUT N,R
```

The machine will then type out

```
YØUR VALUES OF N AND R ARE?
```

Program for ${}_nP_r$ and ${}_nC_r$

```
10 PRINT "YØUR VALUES ØF N AND R ARE":
20 INPUT N,R
30 LET Q=1
40 LET K=1
50 LET Q=Q*K
60 IF N=R THEN 140
70 IF K=(N-R) THEN 120
80 IF K=R THEN 150
90 IF K=N THEN 170
100 LET K=K + 1
110 GØ TØ 50
120 LET T=Q
130 GØ TØ 80
140 LET T=1
150 LET V=Q
160 GØ TØ 90
170 LET P=Q/T
180 LET C=P/V
190 PRINT N,R,P,C
200 GØ TØ 20
210 END
```

Figure T-2

Section I

N! with the Compurer

This section explains how the computer can be used to help you solve factorials. As you have experienced, the factorial of a small number results in a large number. It certainly would be nice if we had a machine to do the series of multiplications that is necessary to get a factorial even for factorials as small as 12! The machine can help us do this.

The first step in preparing ourselves to use the computer would be to draw a flow chart. The purpose of a flow chart is to enable you to see what mathematical operations are done by the computer. You can also see how each step is related to each other step. The flow chart could be thought of as a road map that you would consult in planning a vacation by automobile. A flow chart is not unique. As you could plan several roads to travel during your vacation, so you could plan several flow charts that would compute N factorial. One of these flow charts of N factorial is shown in Figure 1.

The various symbols used in making flow charts each have a special meaning. The parts of the flow chart in Figure 1 on the preceding page are each labeled with a small letter. Refer to Figure 1 for the following explanation.

The symbol at (a) has the sole purpose of showing where to begin using the flow chart.

The symbol at (b) is for input. It tells you what data you need to supply to the computer. The machine takes the number you give it and puts it in a memory location which we will call N. Any time you call for N the machine then goes to this location to find what value you assigned to N. It has many memory locations for storing variables. You may think of a memory location as a sack with a label on it. If we are calculating 7! we would be telling the machine to go to the sack labeled N and put a 7 in it.

The symbol at (c) indicates a computation box. A computation box indicates the instruction we must give the computer. Q = 1 inside the computation box means that we will tell the computer to label a sack Q and to put a 1 in the sack. The left hand side of the equal sign is the label for the memory location (sack), and the right hand side is what is to be put in the location. We ask the computer to put 1 in the Q sack because the first factor in calculating any factorial is 1.

The computation box at (d) means we instruct the computer to put 2 in the memory location marked K. It will become clearer later that we want to put a 2 in this location because we want to calculate 2! first. 2! will be used in the calculation of any higher factorial. We may omit the calculation of 0! and 1! as the result of both of these is 1.

The Q = Q*K is the computation box at (e). It looks confusing until we remember that the Q on the left of the equal sign is simply the label of the sack (memory location), and the Q*K on the right of the equal sign is the instruction; so the statement will mean for the computer to go

Flow Chart N!

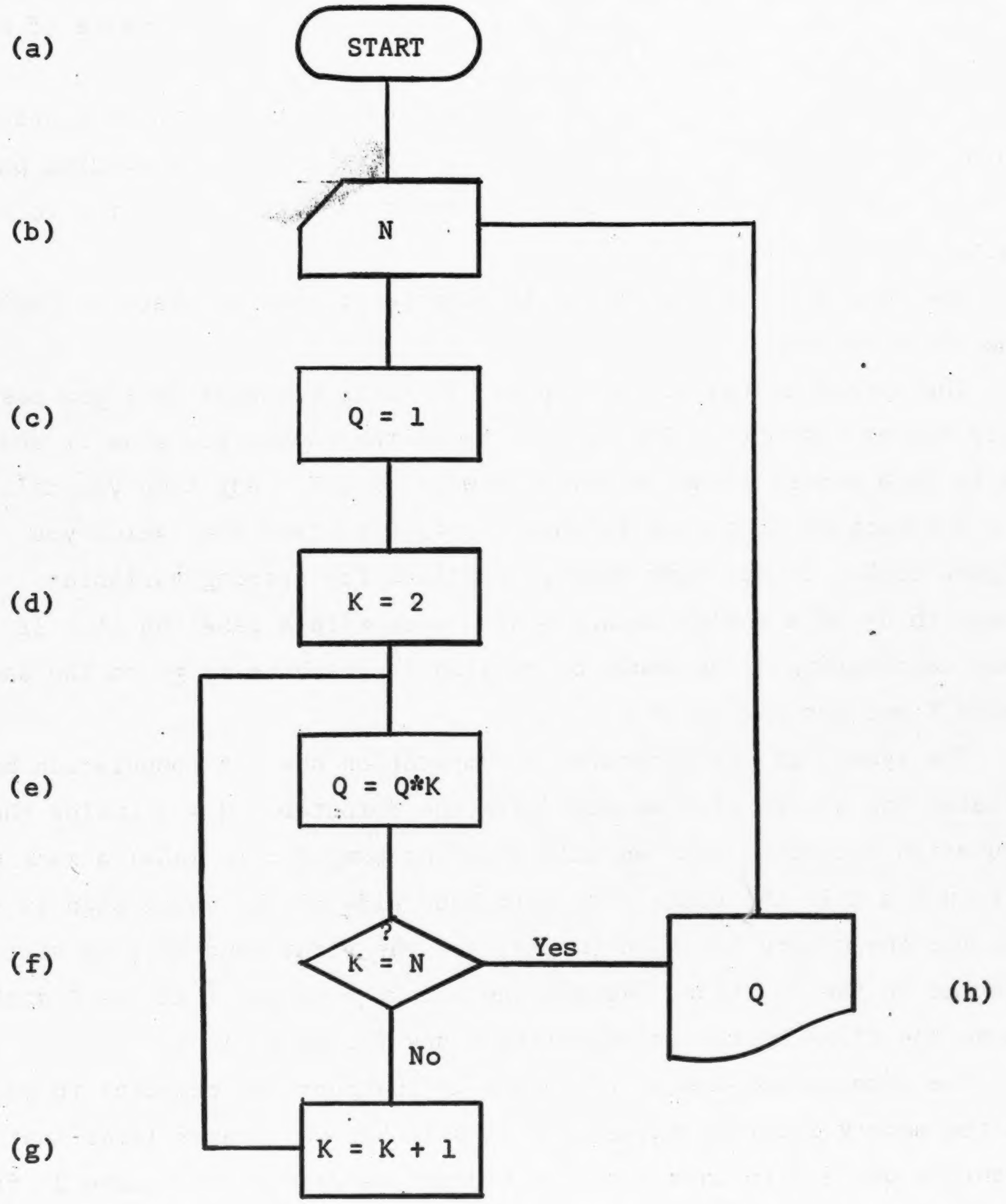

(a)
START
(b)
N
(c)
Q = 1
(d)
K = 2
(e)
Q = Q*K
(f)
K $\overset{?}{=}$ N
Yes
No
(g)
K = K + 1
Q
(h)

Figure 1

to the memory location Q and to the memory location K, then to take the values it finds in the sacks and to multiply (*) the values together. Finally, it will store this new value of Q in the Q sack; and the value of K stored in the K sack remains unchanged at this time.

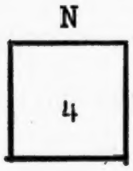
The symbol at (f) means that we will instruct the computer to make a choice, so it is called a decision box. The $K \overset{?}{=} N$ in the decision box means that the computer is to compare the values in the K and N sacks. In other words, the decision box asks the question, "Does K = N?" If the answer is no, the next step in the flow chart is to the computation box at (g). The statement K = K + 1 in that box indicates that the computer will be instructed to go to the K sack, take the value of K that is stored there, increase it by one and put the new value of K back in the sack.

The arrow from the box at (g) goes back to the box at (e). This means that the computer will take this new value for K and multiply it by the value of Q and put the new Q value in the Q sack. This cycle is repeated until the answer to the question "DOES K = N" is yes.

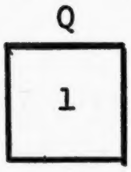
When the answer is yes, the next step is indicated by the symbol at (h). This symbol is the one which is used to indicate that the computer will be instructed to print certain information. In the present example, the computer is directed to print the value of Q which turns out to be the value of N!.

Now take an example to see exactly how it would work. Let N = 4. This means we intend to have the computer calculate 4!. Therefore, 4 is given as the data for the computer to store in the N sack. Now let Q = 1, and let K = 2. The instruction Q = Q*K will give us Q = 1*2 which is 2. This is the value now stored in Q. The decision box now indicates that the computer will compare the value in the K and the N sacks. Since 2 does not equal 4, it will increase K by 1, which is 3. It now computes a new Q which will be 2*3 or 6, where 2 is the last preceding value for Q. Since K is now 3, which is still not equal to 4, K will be increased to 4. The new Q will be 6*4 or 24. When K is compared to N, they will be equal; so the computer will print the value of Q to be 24. This is 4!.

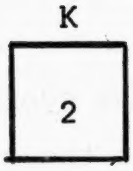
One method of checking to see whether a flow chart will lead to correct results from the computer is to do what is called "playing computer." This involves boxes or squares to represent the memory location of the computer and then following the flow chart to see what happens to the values stored in each location.



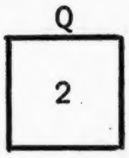
For example, the first data location according to the flow chart shown in Figure 1 is labeled N; so if we supply a value of 4 for N we can represent it as shown.



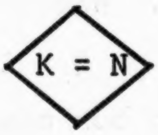
This box represents a value of 1 stored in the memory location labeled Q.



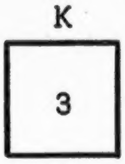
The next step according to the flow chart is to assign a value of 2 in sack K. This is represented as shown.



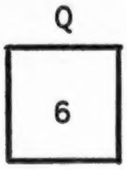
The product of K*Q would now be 1*2 or 2, which is now the value in the sack Q.



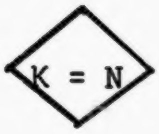
Since K = 2 and N = 4 at this time, the decision box directs the computer to computation box.



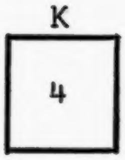
The computation box directs the computer to add 1 to K, which now has a 3 stored there.



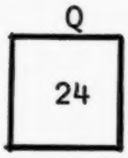
Then the spot labeled Q will have the value of K*Q or 2*3, which is 6.



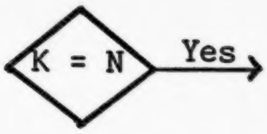
No.



This is obtained by adding 1 to the previous value.

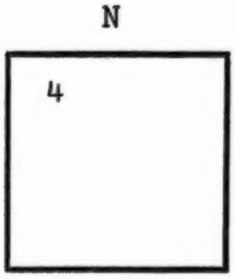


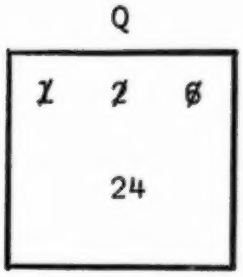
The result of multiplying K times the previous Q.
4*6 = 24.

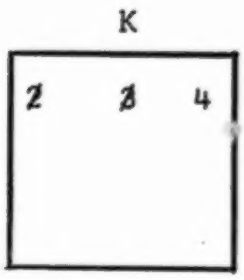


Since K = N, the value of Q is now printed as 24, which is indeed 4!.

In actually "playing computer" the box labeled Q is not redrawn each time a new value for Q is obtained. Instead, the old value is simply crossed out; and the new value is written in. Thus, only one box would be needed to represent each memory location. The results of "playing computer" would then look like the following:







If additional factorials are to be calculated, then a new value would be assigned to N.

The flow chart is not in the form that gives instructions to the computer. Instructions to the computer must be written in a certain way in order for the computer to do the problem. When the instructions indicated in the flow chart are written in proper form, the instructions are called a program. The program for the flow chart shown in Figure 1 might be written as follows:

```
10 READ N
20 LET Q=1
30 LET K=2
40 LET Q=Q*K
50 IF K=N THEN 80
60 LET K=K+1
70 GØ TØ 40
80 PRINT Q
90 DATA 4,8
100 GØ TØ 10
110 END
```

The computer takes your program in numerical order. You should use statement numbers far enough apart so that you can add to your program without rewriting the entire program. For example, if you wish to add another "DATA" statement, you could number the statement 91; and the computer would fit it into your program in proper numerical order.

The "READ" statement tells the machine what variable you want and to set aside a memory location for N. It then goes looking for a "DATA" statement. It now puts 4 in the N sack.

"LET Q=1" tells the computer to set aside a Q sack and put 1 in that sack.

"LET K=2" tells the computer to set aside a K sack and put 2 in that sack.

"LET Q=Q*K" tells it to go to the Q and K sacks. Take the product of the two values, and put this new value in the Q location. Also put the K back in the K location.

"IF K=N THEN 80" has the computer compare K and N, and if they are equal, then to go directly to statement 80; but if they are not equal, it will go to the next statement.

"LET K=K+1" has the machine go to the K location, take K out of the K location, and increase it by one and then put it back in the K location.

"GØ TØ 40" tells it to go back to 40 and to go through again.

"PRINT Q" has it print out the latest value of Q.

"DATA" statement furnishes the values of the variables in the "READ" statement.

"GØ TØ 10" tells the computer to go back to statement 10. Here the computer takes the next value out of the "DATA" statement and assigns that as the new value for N. N now has the value of 8. Now the entire process is repeated again.

"END" tells the computer to quit reading the program.

Section II

Permutation of N Things Taken R at a Time

In the preceding section we learned how a computer could be used in working with permutations. An example was given for the calculation of $_4P_4$ which is the number of permutations of 4 things taken 4 at a time. Since $_nP_n$ is n!, the only thing required of a computer is to calculate n!. Similarly, if we now consider the problem of using a computer to solve $_nP_r$, again the primary operation is the calculation of factorials.

Since $_nP_r$ is $\frac{n!}{(n-r)!}$, the program for the computer results in the calculation of n! and (n-r)! and then divides the results of these two quantities.

The flow chart is given in Figure 2. This is followed by a tentative program for the computer.

Program for $_nP_r$

```
10 READ N,R                         100 LET T=T*K
20 LET Q=1                          110 IF K=(N-R) THEN 140
30 LET K=2                          120 LET K=K+1
40 LET Q=Q*K                        130 GØ TØ 100
50 IF K=N THEN 80                   140 LET Q=Q/T
60 LET K=K+1                        150 PRINT N,R,P
70 GØ TØ 40                         160 GØ TØ 10
80 LET T=1                          170 DATA 5,3,10,2
90 LET K=1                          180 END
```

If we follow the flow chart and the program, it is evident that the first part is simply the calculation of N!, which is the same as the preceding section. The last number to be stored in the memory location labeled Q will be N!. The next part of the flow chart is for the calculation of (N-R)!, and if this section is studied, it should be evident that it is very similar to the first section. It should be clear that the memory location labeled T is being used to store the value of (N-R)! as it is being generated. One difference in the two parts of the program is that in calculating N!, the first step gives 2!, while in the calculation of (N-R)! the first step gives 1!. This is necessary to take care of the cases where R is one less than N. If the program is used to "play computer" for the case of $_5P_3$ the results could be written as follows:

Flow Chart $_nP_r$

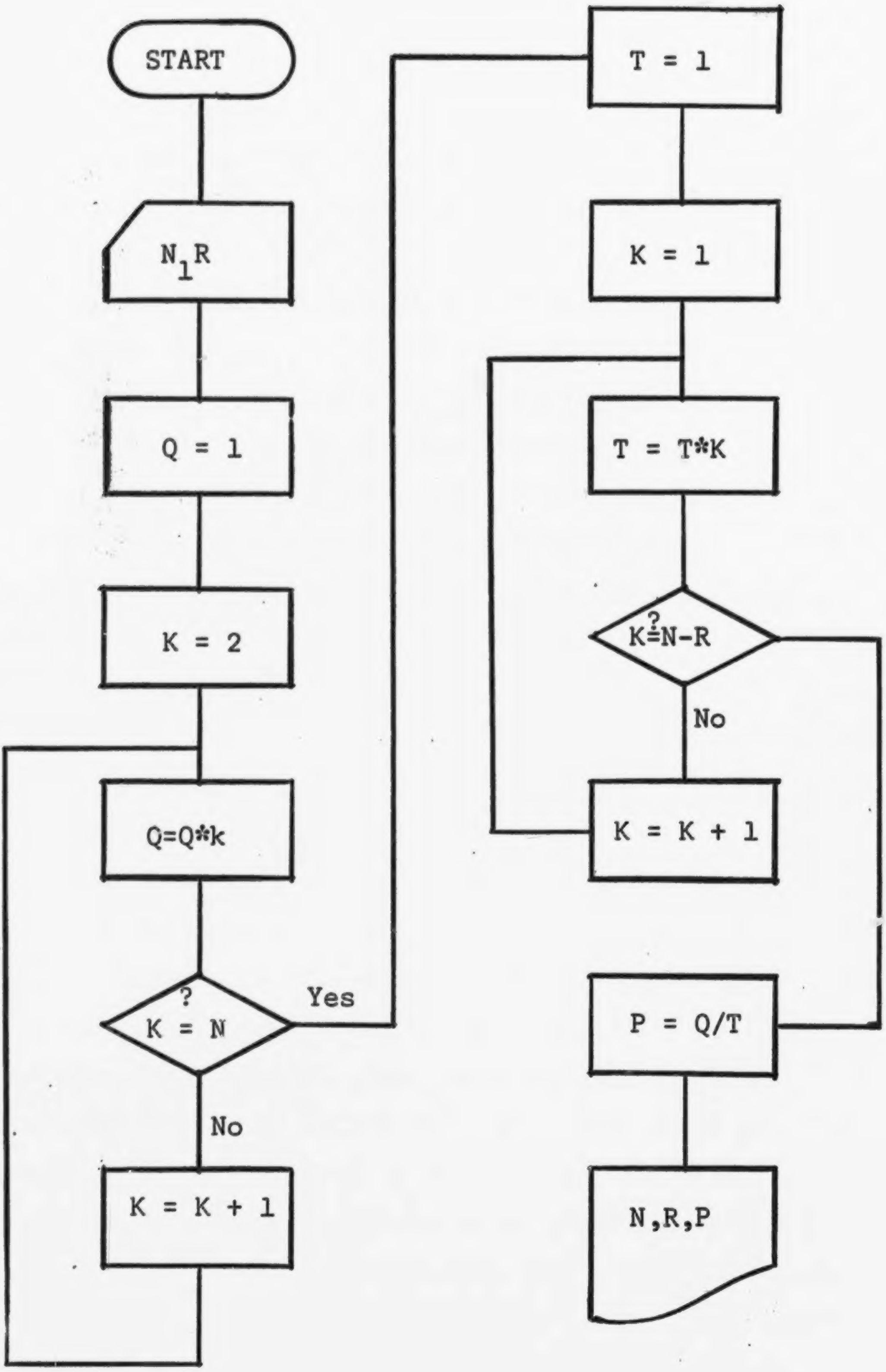

START
N1R
Q = 1
K = 2
Q=Q*k
?
K = N
Yes
No
K = K + 1
T = 1
K = 1
T = T*K
K?N-R
No
K = K + 1
P = Q/T
N,R,P

Figure 2

Line Number	Values in Various Memory Locations N	R	Q	K	T	P
10	5	3	?	?	?	?
20	5	3	1	?	?	?
30	5	3	1	2	?	?
40	5	3	2	2	?	?
50	(GØ TØ 60 because K does not equal N)					
60	5	3	2	3	?	?
70	(GØ TØ 40)					
40	5	3	6	3	?	?
50	(GØ TØ 60 because K does not equal N)					
60	5	3	6	4	?	?
70	(GØ TØ 40)					
40	5	3	24	4	?	?
50	(GØ TØ 60 because K does not equal N)					
60	5	3	24	5	?	?
70	(GØ TØ 40)					
40	5	3	120	5	?	?
50	(GØ TØ 80 because K does equal N)					
80	5	3	120	5	1	?
90	5	3	120	1	1	?
100	5	3	120	1	1	?
110	(GØ TØ 120 because K does not equal N-R)					
120	5	3	120	3	1	?
130	(GØ TØ 100)					
100	5	3	120	2	2	?
110	(GØ TØ 140 because K does equal N-R)					
140	5	3	120	2	2	60

At this point the next instruction to the computer is to print the values of N, R and P. We see that P has the value of 60, which is $\frac{n!}{(n-R)!}$ or $\frac{5!}{(5-3)!}$.

This leads us to believe that the program is all right as it is written. If "playing computer" shows that incorrect results are obtained, then the flow chart and program must be corrected accordingly. If a computer were actually available, the next step would be to run the program on the computer to see if it would produce the correct results.

If additional permutations were to be calculated, then new values would be assigned to N and R.

The question might arise as to whether the program for the calculation of $\frac{N!}{(N-R)!}$ could be used to find the value of ${}_nP_n$, i.e., the case where R is equal to N. In this case, N - R would be zero. If we look at the steps obtained in "playing computer" for ${}_5P_3$ and we come to line 110 the first time and then compare K with N - R, we see that K is already greater than N - R. Consequently, they could never be equal. Thus, K would continue to become one greater, and no solution would be reached. The next question that might occur is whether or not the flow chart and the program might be modified to work for the case where "N = R." One possibility would be to change the question in the second decision box to "K $\geq$ (N-R)" instead of "K=(N-R)." Line 110 would then read "IF K $\geq$ (N-R) THEN 140." Check to see if it would produce the desired result.

Section III

The program for $_nC_r$ might be written as follows:

```
10 READ N,R
20 LET Q=1
30 LET K=2
40 LET Q=Q*K
50 IF K=N THEN 80
60 LET K=K+1
70 GØ TØ 40
80 LET T=1
90 LET K=1
100 LET T=T*K
110 IF K=(N-R) THEN 140
120 LET K=K+1
130 GØ TØ 100
140 LET V=1
150 LET K=1
160 LET V=V*K
170 IF K=R THEN 200
180 LET K=K+1
190 GØ TØ 160
200 LET C=Q/(T*V)
210 PRINT N,R,C
220 DATA 5,2
230 END
```

The flow chart and the program for $_nC_r$ are tentatively correct. This is the combination of n things taken r at a time. It can be seen that they are the same as the one for $_nP_r$ up to line 120. The remaining part is for the calculation of R; and of C = Q/(T*V).

It is suggested for the student to "play computer" for the values of N = 5 and R = 2.

It should be pointed out that flow charts and programs are not unique for any given problem. That is, there are probably a number of different programs which would produce the desired results. For example, in the program written for $_nC_r = \frac{N!}{(N-R)!R!}$ the procedure for calculating a factorial was used three times. Somewhere in the calculation of N! the computer would calculate an intermediate value which would be equal to (N-R)!. Also, the value of R! would be obtained during the calculation of N!. It might be possible to write a program which would select these intermediate values and store them for use in calculating $\frac{N!}{(N-R)!R!}$.

It would be worthwhile for students who are interested in doing more with the computer to write a flow chart developing the idea in the preceding paragraph.

Flow Chart ${}_{n}C_{n} = \frac{N!}{(N-R)!R!}$

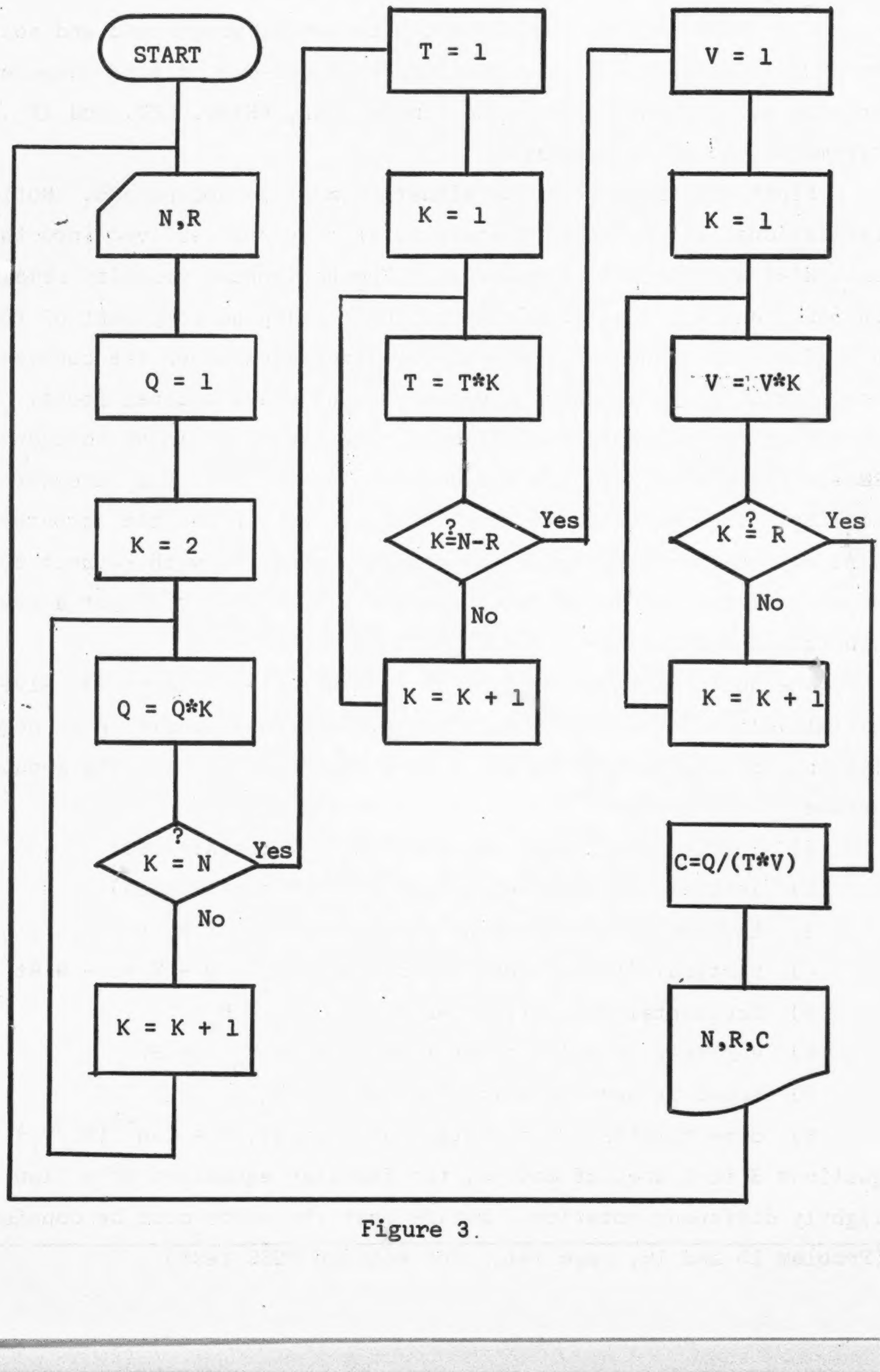

START
N,R
Q = 1
K = 2
Q = Q*K
K $\overset{?}{=}$ N
Yes
No
K = K + 1
T = 1
K = 1
T = T*K
K $\overset{?}{=}$ N-R
Yes
No
K = K + 1
V = 1
K = 1
V = V*K
K $\overset{?}{=}$ R
Yes
No
K = K + 1
C=Q/(T*V)
N,R,C

Figure 3.

THE SOLUTION OF A TYPICAL PROBLEM CONCERNING MOTION OF OBJECTS IN A GRAVITATIONAL FIELD

by Marion E. Hakewessel

A ball is thrown into the air with an initial velocity V at an angle A with the horizontal at a height D above the ground.

a) Plot the trajectory of the ball using k second intervals until it hits the ground.

b) Determine the magnitude and direction of the velocity of the ball at the above k second intervals.*

A problem such as the above can be easily programmed and solved by students. One must assume a preliminary knowledge of some computer language such as BASIC; only the simple READ, PRINT, LET, and IF ... THEN statements should be required.

First the physics of the situation must be understood. Motions in a gravitational field are best analyzed if they are resolved into their horizontal and vertical components. The horizontal velocity remains constant, and acceleration is zero, their being no component of force in this direction. The vertical velocity increases under the constant acceleration $g = -9.8 \text{m/sec}^2$ due to constant gravitational force. Thus we can set up two systems of equations: one for determining horizontal displacement and velocity; one for determining vertical displacement and velocity. Then at selected time intervals we can ask the computer to print out the coordinates of the object's position with respect to a ground position and to add the velocities vectorially to get a resultant magnitude and direction.

The suitable equations are as follows: (See Figure 1) Given initial velocity V in m/sec at angle A in degrees (positive or negative) with the horizontal at a height D in m from (above only) the ground surface.

1) initial horizontal velocity: $H_o = V \cdot \cos(A:\pi/180)$

2) initial vertical velocity: $V_o = V \cdot \sin(A \cdot \pi/180)$

3) horizontal displacement after t sec: $D_1 = H_o \cdot t$

4) vertical displacement after t sec: $D_2 = D + V_o \cdot t - 9.8t^2/2$

5) horizontal velocity after t sec: $V_1 = H_o$

6) vertical velocity after t sec: $V_2 = V_o - 9.8t$

7) speed at any instant: $S = (V_1^2 + V_2^2)$

8) direction(from horizontal) at instant: $N = \tan^{-1}(V_2/V_1) \cdot 180/\pi$

Equations 3 to 6 are, of course, the familiar equations of motion with slightly different notation. Notice that the units must be consistent.

---

*(Problem 13 and 16, page 341, 1960 edition PSSC text)

Notice, too, that the angles must be converted to radian measure before using the BASIC functions SIN and COS; also a conversion back to degree measure must be made if the BASIC arctangent function is used.

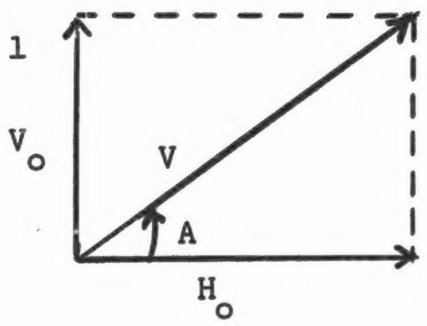
The next step is to sketch a flow chart to help understand the logic of the program. Refer to Figure 2. In the first box we read in the givens. Then we calculate the initial horizontal and vertical velocities. It is at this point that we want to "time" the flight; the computer will print all the variables at t=0, then recompute them a time increment k sec later. It will continue this loop until the object "hits" ground (vertical displacement is less than zero).

A program that follows the flow chart in BASIC language is in Figure 3.

The results of a run using sample data V = 10m/sec, A = 45°, D = 0 m, k = .1 sec, is shown above. Notice the following: When the path begins to point down, the value of the angle from the horizontal begins to be negative. Also notice one negative value for vertical distance; the object apparently "hit" the ground between 1.4 and 1.5 sec. The conditional statement (line number 140) allows the computer one more calculation as long as the previous height above ground is not less than zero. Thus, if the values are plotted, the place where the object "hit" could be read from the graph. Students should also realize that the values for speed are close to proportional to the lengths of the velocity vectors whose direction is given. The plotted points are closer together for slower speeds ... like the images on a "multiple flash" photograph.

The program could be modified to include calculation of the exact time of hitting the ground, the highest point on the trajectory, exact "range", etc. If it were desired to generalize the program to include cases where the initial position of the object was "below" the ground, an additional IF ...THEN loop would have to be included to determine whether any positive values of D_2 have yet been found. (There is always the possibility that the object never gets "up to" the ground surface.) Of course this program is valid only "near" the earth's surface where g is a constant. If the value of g as a function of height above the earth were substituted for -9.8, then the program could be further generalized. For example, define the following: `15 DEF FNG(D2) = .667E-11 * M/(D2*D2)` where M is the mass of the earth and D_2 is the distance from the center of the earth. Use FNG(D2) in place of a constant value of g.

Fig. 1


V
V_o
A
H_o

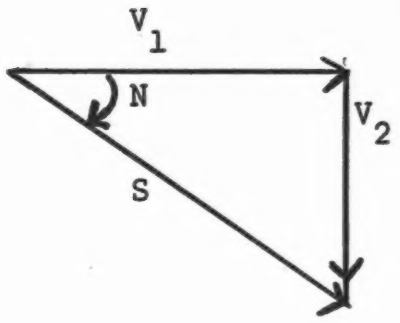
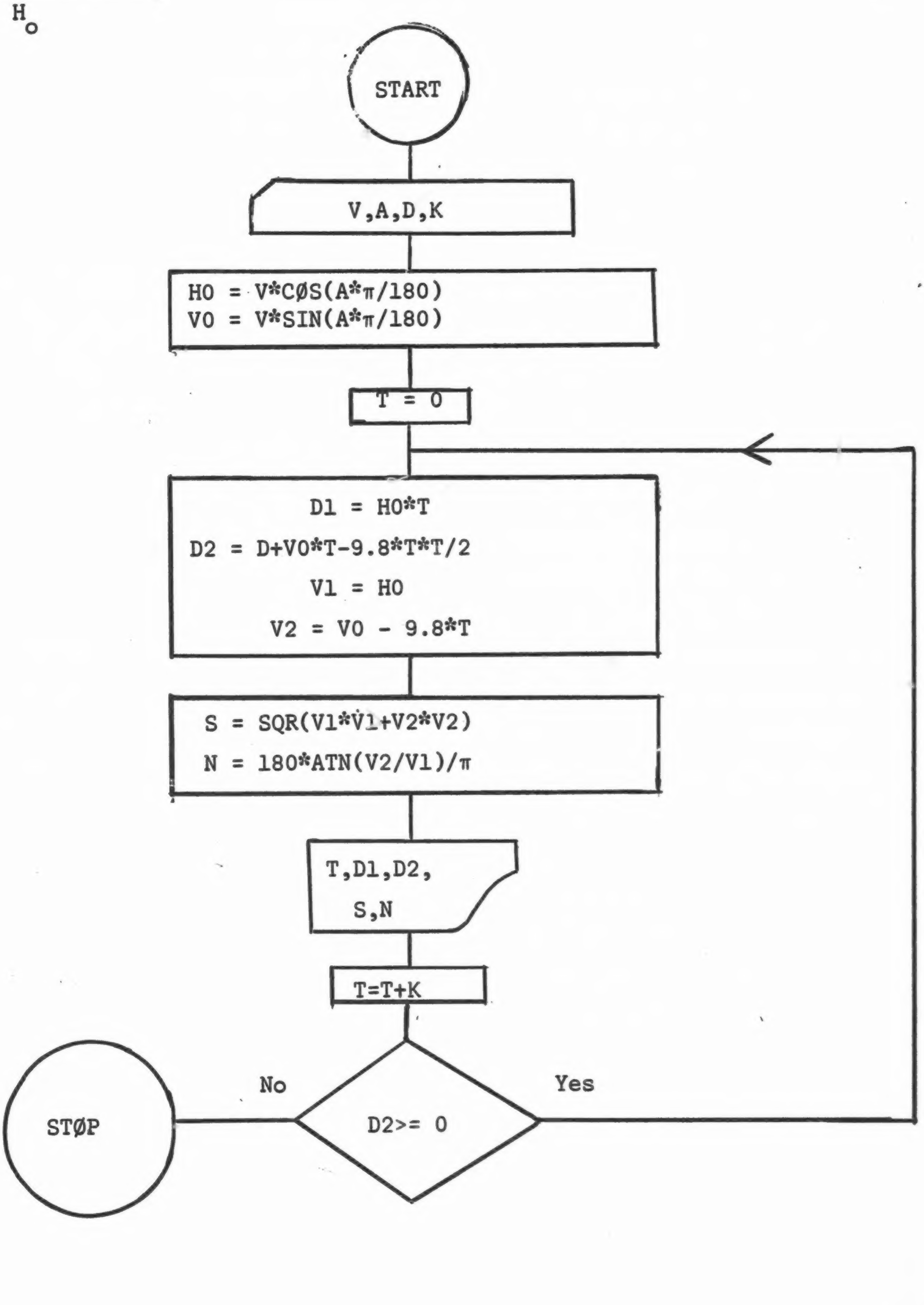

V_1
N
V_2
S

Fig. 2


START
V,A,D,K
HO = V*CØS(A*π/180)
VO = V*SIN(A*π/180)
T = 0
D1 = H0*T
D2 = D+V0*T-9.8*T*T/2
V1 = H0
V2 = V0 - 9.8*T
S = SQR(V1*V1+V2*V2)
N = 180*ATN(V2/V1)/π
T,D1,D2,
S,N
T=T+K
D2>= 0
No
Yes
STØP

```
TRAJEC    16:13     PX MØN 08/14/67

TIME             HØRIZ D          VERT D           SPEED            DIRECTION
 0               0                0                10.              45.
 .1              .707105          .658108          9.3328           40.7419
 .2              1.41421          1.21822          8.72485          35.86
 .3              2.12132          1.68032          8.18936          30.2944
 .4              2.82842          2.04443          7.74139          24.0192
 .5              3.53553          2.31054          7.39685          17.0684
 .6              4.24263          2.47865          7.17067          9.56139
 .7              4.94974          2.54876          7.0742           1.70985
 .8              5.65684          2.52086          7.11274         -6.20605
 .9              6.36395          2.39497          7.28413         -13.8924
 1.              7.07105          2.17108          7.57937         -21.103
 1.1             7.77816          1.84919          7.98473         -27.6779
 1.2             8.48527          1.4293           8.48444         -33.5489
 1.3             9.19237          .911405          9.06292         -28.7193
 1.4             9.89948          .295513          9.70608         -43.2376
 1.5             10.6066         -.418379          10.4019         -47.1732
```

Fig. 3

```
 10 READ V,A,D,K
 20 PRINT "TIME","HØRIZ D","VERTD","SPEED","DIRECTIØN"
 30 LET HO= V*CØS(A*3.1416/180)
 40 LET VO = V*SIN(A*3.1416/180)
 50 LET T = 0
 60 LET D1 = HO*T
 70 LET D2 = D + VO*T - 9.8*T*T/2
 80 LET V1 = HO
 90 LET V2 = VO - 9.8*T
100 LET S = SQR(V1*V1 + V2*V2)
110 LET N = 180*ATN(V2/V1)/3.1416
120 PRINT T,D1,D2,S,N
130 LET T = T + K
140 IF D2> = 0 THEN 60
150 PRINT
160 GØTØ 10
170 DATA 10,45,0,.1,12.6,37,6.9,.2
180 END
```

THE USE OF THE COMPUTER IN FINDING THE APPROXIMATE VALUE OF PI

by Harry A. Hurley

Time after time while teaching math, the writer had been asked by students, "How are we able to say that Pi has an approximate value of 3.14158470+?" or, "How do we know that the ratio of the circumference of a circle to its diameter is approximately 3.14158470+?" It was therefore decided to use the computer in an attempt to answer these questions. This was not a new computation, but by the use of the computer it was thought it could be demonstrated; and the task of computing each step could be shortened.

By geometric proof, it could be shown that, given a regular inscribed polygon of n sides of s length, the length of a side of a regular inscribed polygon of 2n sides is equal to $\sqrt{2r^2 - r\sqrt{4r^2 - (s_n)^2}}$

Graphically, it was easy to show that as the number of the sides of the regular inscribed polygon was doubled, the closer the perimeter of the polygon came to the circumference of the circle. The above formula was used in computing the length of a side of each new polygon. A hexagon was used as the polygon to start with, and since 1 was used as the radius, 1 was the length of the side of the first polygon.

To do this manipulating and computing by hand would have been a gargantuan job; the computer did it quickly.

As stated above, to compute the value of Pi, a regular inscribed polygon of six sides was used as the first for our computations. The formula stated above was used to determine the length of the side of a regular polygon of 2n or 12 sides. The length of a side was multiplied by the number of sides to obtain the perimeter of the polygon. From this the ratio of the perimeter to diameter was computed. The following results were obtained:

No. of Sides	Length of Side	Perimeter	Approximate Value of Pi
6	1	6	3
12	0.517638	6.21166	3.10583
24	0.261052	6.26526	3.13263
48	0.130806	6.2787	3.13935
96	6.54382 E-2	6.28207	3.14103
192	3.27235 E-2	6.28291	3.14146
384	1.63624 E-2	6.28315	3.14158
768	8.18144 E-3	6.28335	3.14167

The above table showed that when the number of sides increased beyond $n \times 2^6$ the approximation went over the value of Pi. The reason for this was that the computer rounded off to the nearest 8th decimal place. This caused an increase in the error which the machine could not correct. By chopping manually, it was possible to carry the computations two steps further to an increase in the number of sides to 768, 1536 and, by a drastic chopping, 3072 and 6144.

For a check to determine if there was an error created by the choice of the original number of sides, the same program was run where n = 4. The following results were obtained. The manual chopping was not performed on this running of the program.

No. of Sides	Length of Side	Perimeter	Approximate Value of Pi
8	.765358	6.12286	3.06143
16	.390176	6.24282	3.12141
32	.196032	6.27302	3.13651
64	9.81342 E-2	6.28059	3.14029
128	4.90819 E-2	6.28248	3.14124
256	2.45428 E-2	6.28296	3.14148
512	1.22717 E-2	6.28312	3.14156

This showed that there was no error in the number of sides chosen as a start, for the computations became too great at the same step.

Given a side of a regular inscribed polygon of n sides, to find the side of a regular inscribed polygon of 2n sides, see Figure 1.

Given: ⊙ O with radius r, $\overline{AB}(s)$ a side of a regular inscribed polygon of n sides, and $\overline{AC}(x)$ a side of a regular inscribed polygon of 2n sides.

To Find: the value of x.

Solution:

Statements	Reasons
1. Draw $\overline{CO}$ intersecting AB at D	1. Two points determine a line.
2. $\overline{AC} = \overline{BC}$	2. Def. of regular polygon
3. $\overline{AO} = \overline{BO}$	3. Radii are =
4. CO is ⊥ bisector of $\overline{AB}$	4. If two points are equidistant from the end points of a line segment in their plane and are on opposite sides of the line segment, then they determine the perpendicular bisector of the line segment.
5. $\overline{AD} = \frac{S}{2}$	5. $\overline{CO}$ is ⊥ bisector of $\overline{AD}$.
6. $\overline{OD} = \sqrt{r^2 - \frac{S^2}{4}}$	6. Pythagorean theorem
7. $\overline{CD} = r - \sqrt{r^2 - \frac{S^2}{4}}$	7. Subtraction
8. $x = \sqrt{\frac{S^2}{4} + r - \left(\sqrt{r^2 - \frac{S^2}{4}}\right)^2}$	8. Pythagorean theorem

$$x = \sqrt{\frac{S^2}{4} + r^2 - 2r\sqrt{r^2 - \frac{S^2}{4}} + r^2 - \frac{S^2}{4}}$$

$$x = \sqrt{2r^2 - 2r\sqrt{r^2 - \frac{S^2}{4}}} \qquad x = \sqrt{2r^2 - r\sqrt{4r^2 - S^2}}$$

The proof of the above formula was left to the students. To show how great a task the complete computation would be, the students were asked to carry out the computation of the approximation of Pi for a step or two.

The flow chart for this computation is shown in Figure 2.

<u>Program for Computing Approximate Value of Pi</u>

```
10   READ S,N
20   PRINT "N","S","C","P"
30   FOR K = 1 TO 10
40   LET S = SQR(2-SQR(4-S*S))
50   LET N = 2*N
60   LET C = S*N
70   LET P = C/2
80   PRINT N,S,C,P
90   NEXT K
100  DATA 1,6
110  END
```

So that the lopping could be done by hand, the following changes were made in the above program:

```
10   INPUT S,N
30   (TAKE ØUT)
90   (TAKE ØUT)
100  (TAKE ØUT)
```

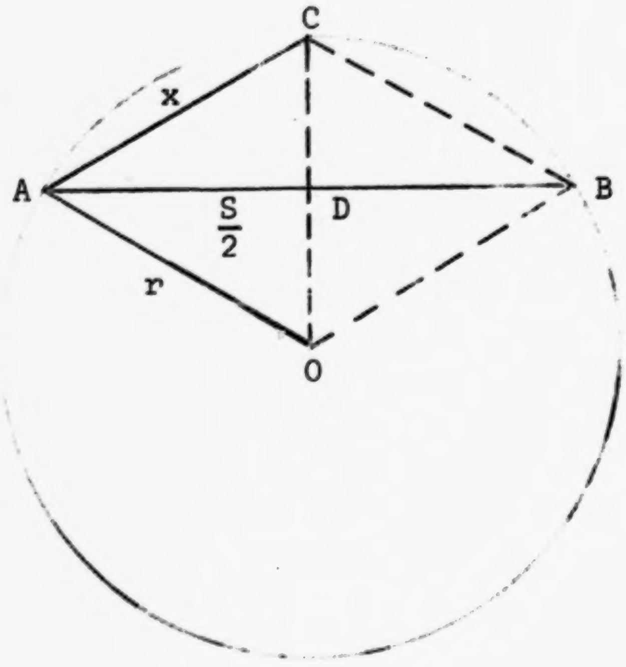


Figure 1

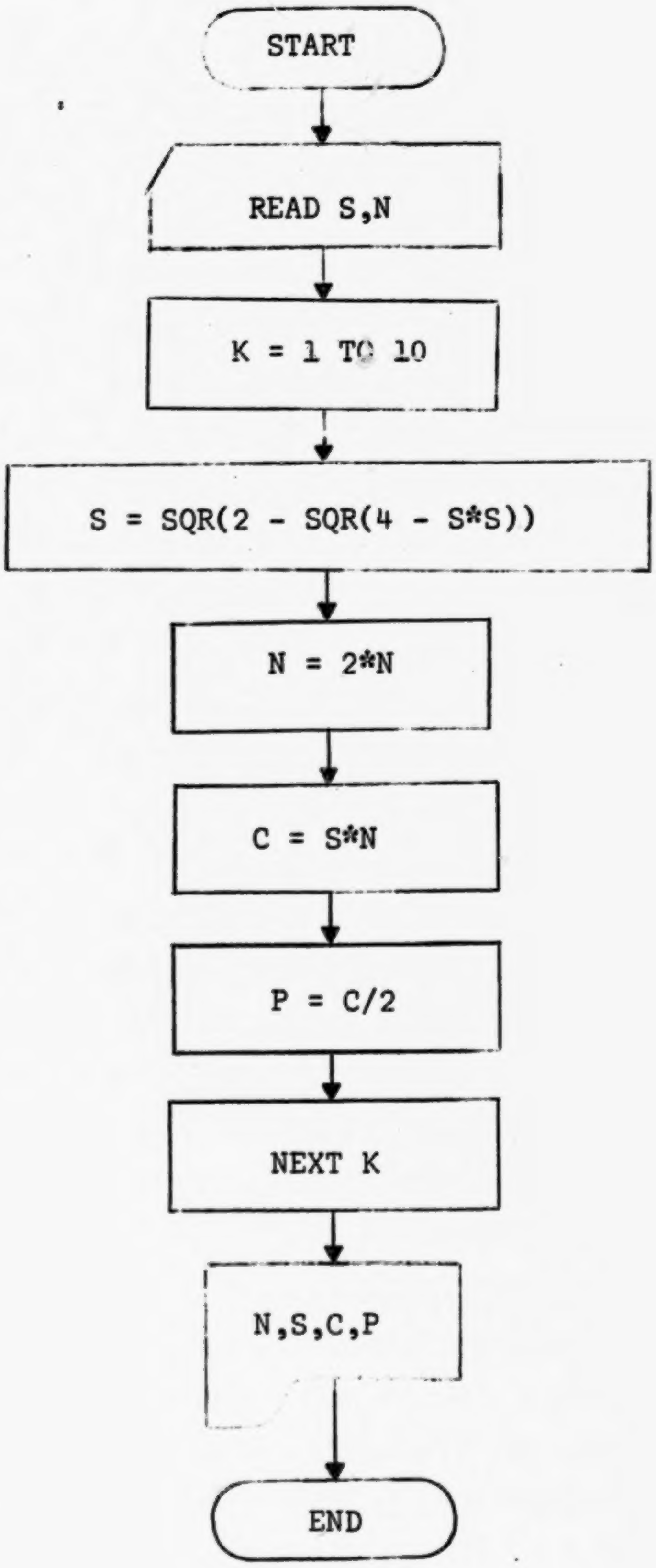

START
READ S,N
K = 1 TO 10
S = SQR(2 - SQR(4 - S*S))
N = 2*N
C = S*N
P = C/2
NEXT K
N,S,C,P
END

Figure 2

CONSTRUCTING FLOW CHARTS AND WRITING PROGRAMS FOR SIMPLE PROBLEMS

BY Dan H. Jarrell and Royce D. Jackson

CONTENTS

I. INTRODUCTION
    A. Computer language
    B. Time sharing systems
    C. Flow charts
II. Construction of flow charts
    A. Symbols
    B. Loops and indexes
    C. Examples
    D. Exercises
III. Writing computer programs
    A. Loops and print statements
    B. Writing BASIC programs from flow charts
        1. Examples
        2. Discussion of approaches to improve programs
    C. Exercises
IV. Expanding computer applications
    A. Formulas or mathematical models
    B. Examples
        1. Application in geometry relating circles
        2. Simple series
        3. Game of twenty-three matches

We are presently living in the early stages of the age of the computer. The public has a wide range of ideas concerning the computer. Man has devised languages in order that he may communicate with the computer. Computer procedures for the solution of mathematical problems are usually written in an algorithmic language, a language that computer scientists have especially designed for expressing problem-solving procedures. The most commonly used algorithmic languages are ALGOL (Algorithmic Language) and FORTRAN (Formula Translator).

In this discussion we will describe a language called BASIC, which is a modified form of ALGOL. BASIC is designed especially for instructional use, it is simpler and easier to learn than ALGOL or FORTRAN.

For educational purposes, a computer for each school is financially impossible at the present time. Thus, the Time Sharing System is presently being used for this purpose. This system consists of a large central computer that has many input-output stations, all of which can be in use at the same time. This means that a number of individuals can simultaneously use the computer on a "shared-time" basis.

Each school is equipped with a teletype. The teletype is connected by long distance telephone lines to the computer. The teletype itself is similar to a regular typewriter.

Basically we use the time sharing system in the following manner:

1. Write a flow chart and then write the problem solving procedure in BASIC language.

2. Type the program on the teletype without being in contact with the computer. The teletype will punch the program on tape.

3. Then call the computer. The teletype is equipped with a system similar to a telephone. After getting in contact with the computer, run the prepared tape through the teletype tape reader, storing the program in the computer.

4. Direct the computer to run the procedure. The computer will run the procedure and type the results on the teletype.

5. Debug the procedure, if necessary, and try again.

Therefore, we must first of all learn to write problem procedures in the BASIC language and learn to use the teletype to communicate with the computer.

In our discussion that follows the reader will gain some knowledge in using "Basic Language". However, it will be necessary to consult reference manuals for a more advanced study.

Introduction to Flow Charts

Since it is evident that man-to-computer communication must be concise, consistent, and complete to yield a satisfactory relationship, it also follows that the analysis and interpretation of a problem must have the same characteristics to facilitate its translation into a workable program for the computer. A logical sequence of steps that can be understood and acted upon by a computer must be followed to gain a satisfactory solution. This sequence is obtained by a detailed analysis of the relationships found in a particular problem. Such an analysis may well be done by using a "flow chart" to indicate the required steps. A flow chart is a pictorial display indicating the chain of steps involved in the solution of a problem. Such a chart presents an overall view of a procedure that is frequently much easier to group than a description in words or in ordinary algebraic expressions. In addition to the sequence of steps involved, the chart includes conditional and repeating loops which give directive instructions at certain points in the sequence of steps which are vital to the efficiency of the total solution procedure. Usually the more complex problems require a more complex flow chart.

This study in computer programming is primarily concerned with BASIC language, which adapts quite readily to program translation from flow charts. However, flow charts can be readily adapted and utilized in conjunction with other languages and different models of computers. Symbols denoting specific operations or considerations in procedure have been standardized to a great degree to promote and facilitate universal acceptance and utilization of flow charts. This symbolic language is also understood by mathematicians who may not be familiar with any computer language but who utilize their skills to analyze complex problems and forward the analyses by flow charts to a programmer.

Once a flow chart is constructed, the major part of the total procedure in solving a problem by a computer is completed. A good flow chart gives sufficient details in information to write a workable computer program. No step that is necessary to a solution may be skipped or placed out of its proper sequence in a chart. Logic, proceeding step by step and in order, is vital to successful problem solving. Mathematics is a science based on organization and procedure. Construction of flow charts and writing computer programs can be most useful tools in teaching organization and procedure in problem solving and in making more concrete a student's understanding of the fundamental arithmetic operations, effective manipulations of terms and expressions, and recognition of relationships between quantities.

Flow Chart Symbols

Symbols used in construction of flow charts are illustrated and explained below. A few symbols are adequate to portray most problems that yield themselves to computer application.

1. The beginning of the program and points at which the computer is started are indicated by an oval.

2. Points at which data values are either read into the computer or printed out as the computer asks for information are indicated in and enclosed by a pentagon resembling a rectangle with the top left-hand corner clipped off. (It is given the familiar shape of the IBM card.)

3. Statements instructing the computer to calculate a certain quantity or to set it equal to another quantity are enclosed in a rectangle.

4. A point at which a decision is made and the program must choose between two or three branches to other parts of the program is indicated by conditional statements enclosed in a diamond. The appropriate branch is followed depending on the answer to the question or conditional restrictions in the decision box.

5. If a flow chart is too large to fit on a single page or column, or if a branch line is difficult to draw between two points, a letter enclosed in a small circle serves as a connection link between sections.

Loops and Indexes

Loops in a flow chart are indicated by lines connecting parts out of their natural sequence, using arrows to denote the direction and to what point the computer should go to resume operation on the program. Repeating loops instruct the computer to repeat a sequence of steps that may include only a few steps or the entire program. An example of the latter is this: When one set of data is utilized in the program and an answer printed, it may be desired that a new set of data be entered and processed through the entire program. To accomplish this a loop is set up to direct the computer to the input statement after completion of the steps using the initial set of data. Conditional loops are used in conjunction with decision (diamond) boxes and may also be used with repeating loops.

Indexes are used as counters or as subscripts to variables to enable counting the number of passes through a particular loop and/or to differentiate between variables as they are read in or calculated. Loops and indexes are illustrated in the examples listed following this section. In these examples of problems shown, certain descriptive or indicative words such as "let, calculate, add, input, and output" included in the boxes are normally left out in constructing flow charts. These are used to familiarize the student with the function of the symbols used. Succeeding examples and exercises will delete these words since the symbols used indicate the type of operation or statement. The examples illustrated do not necessarily indicate the most efficient procedures; they are intended illustrate ways in which the symbols may be used in a program.

Exercises: Construct flow charts to represent the procedures in solving the following for Y, V, or A.

A1. $y = 2a-b$

A2. $A = \pi R^2$

A3. $V = 1/3\pi R^2h$

A4. $3ax = 2d$

A5. $1/3y - b = 5c$

A6. $2A = (b_1 + b_2)\ h$ (set up a loop to repeat the program for more than one set of values for b_1, b_2, & h)

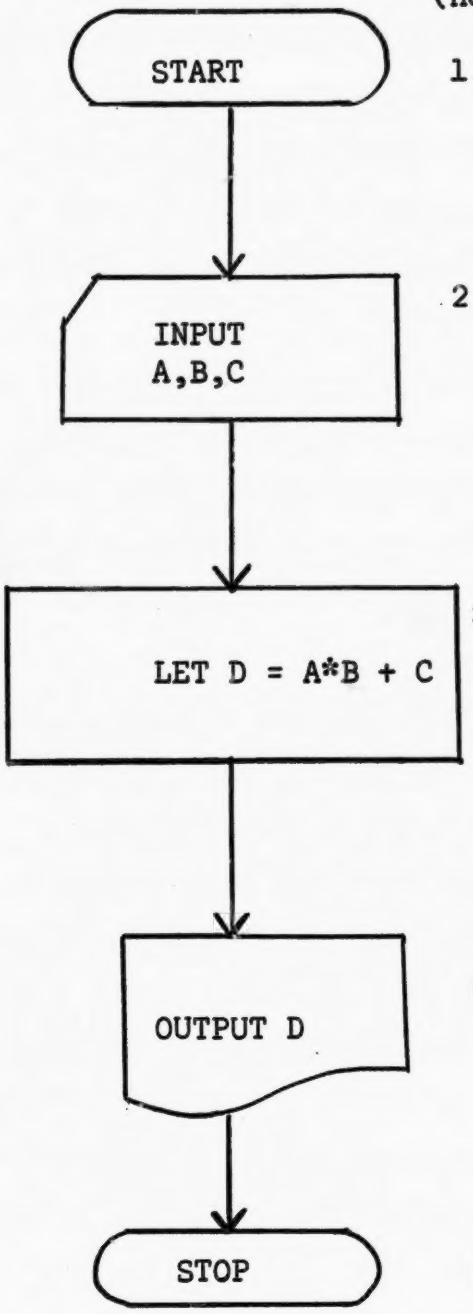
A7. $y_i = 2B_i^2$ (set up loops to read 5 values of B and subscript both B and Y)

A8. $BY = A - C$ (set up a conditional loop to avoid dividing by zero and another loop to use more than one set of data for A,B, & C)

A9. $y = AX^2 + BX + C$ (find points (x,y) to enable the drawing of the graph of the given function)

A10. $y_i = 3C^2$ (write a statement to let $C = C + 1$ below the given statement, print out y_i, and set up a loop repeating the calculation of y_i for 5 different values of C)

Example A1: Given: $d - c = ab$ and values for a, b, & c solve for d (no loops are included in this example)



1. The oval indicates the starting point of the program.

2. To solve for d, values for a,b, & c must be given. These are either read in from a "data" line or given as input information as requested by the computer.

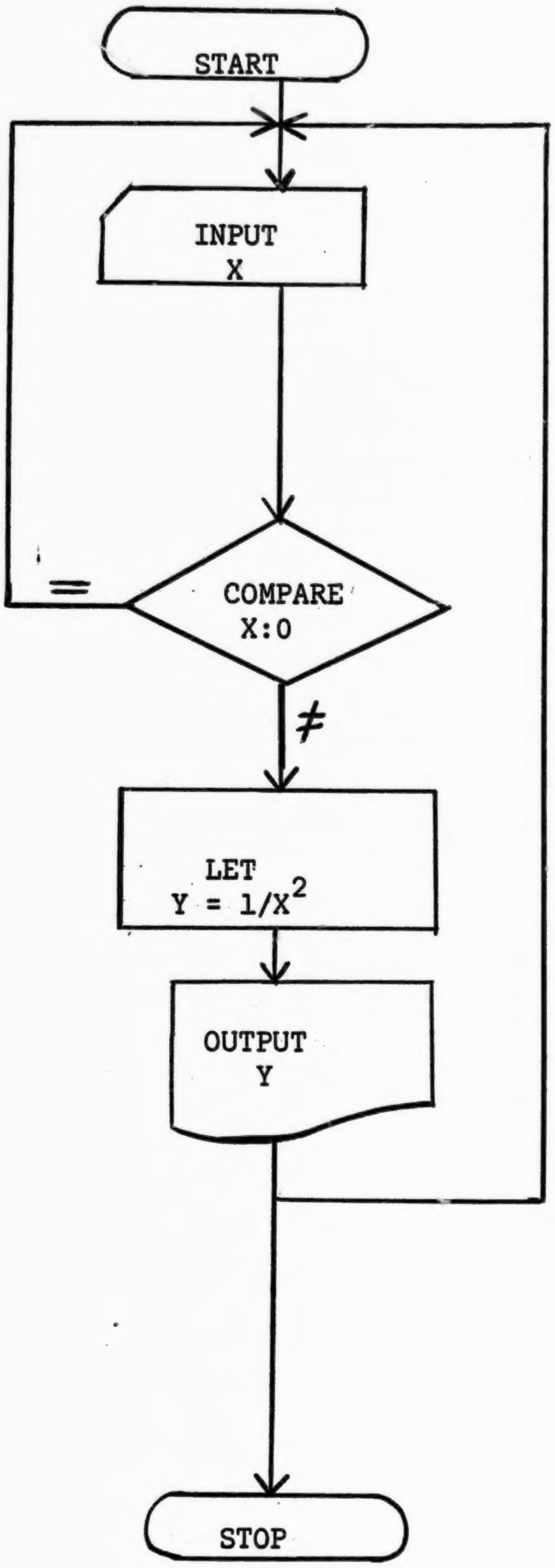
3. The desired quantity d is equated to the proper expression requiring rewriting the given equation in terms of d, and calculation is made.

4. The computer prints out the calculated value for d.

5. The sentence is completed so the program is terminated.

Example A2: Given $y = 1/X^2$, solve for y

A conditional loop and a repeating loop are used here.



1. The program is started.

2. Values for X are read from the data line or values are input as requested.

3. The computer is instructed to compare the input value of X with zero. If it is equal to zero (the next step shows that this would mean dividing by zero, which is not permitted) a loop instructs the computer to return for another input value of X. If the comparison shows that X is not equal to zero then the decision box directs to the next step in the sequence.

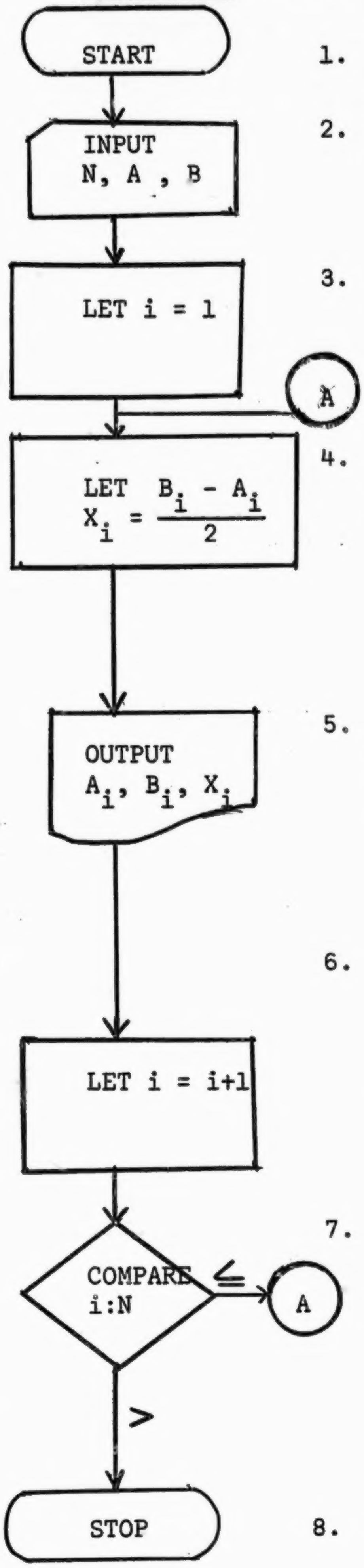
4. y is set equal to the indicated quantity and a calculation is made.

5. The calculated value for y is printed.

6. A loop is set up to direct the computer to the beginning of the program where a new value of X is taken and the entire program sequence is followed as before.

7. When all values of X are read from the data line, or input values are terminated, the computer is instructed to terminate operations.

Example A3: Given $X = \frac{B - A}{2}$, solve for X for N values of A and of B



1. Start the program.

2. Again values are input as requested or are read in from data lines. N is used to limit the counter i.

3. "i" is used to count the number of passes through the sequence and also as a subscript to X,A, & B

4. "X" is set equal to the indicated quantity and calculated. It is also subscripted to differentiate between the various values calculated corresponding to the different input values of A and B.

5. Here the values of A and B corresponding to the calculated value of X are printed as well as the X value. This enables correlating the calculated X value with the proper A and B values by inspection of the output since they are now grouped.

6. This is not an ordinary statement of equality but a statement meaning that the value of i is to be changed by adding 1 to whatever value i held before reaching this point. If this is the first pass through the loop, then i held the value of 1 and so 1 is now added to that, giving i a new value of 2. Each pass through the loop at this point increases the value of i by 1.

7. In the decision box i is compared to N. If $i \leq N$, then a repeating loop is set up between the decision box and Ⓐ, and this loop is repeated until the value of i exceeds N. The program then proceeds to the next and last step.

8. The program is now terminated.

Loops and Print Statements in BASIC Programs

In writing BASIC programs, if the flow chart contains loops, the program will also contain loops. There are several ways to set up the types of loops used. A conditional loop may use an "if-then" statement. "If" certain conditions are met, "then" the computer is directed to a specific point to resume operation on the program or to terminate it. A type of repeating loop is the "for" and "next" statements. The "for" statement introduces the loop and a "next" statement at the end of the loop directs the computer to the beginning of the loop for another value of the variable used as a counter and/or a subscript. The "for" statement sets the bounds on the variable. Passes are made through the loop until all values of the variable, up to and including the limiting value, have been used. Any number of these loops may be used in the same program. A loop may be entirely inside another such loop, but they must not intersect. The inner loop must be completed first before reaching the "next" statement that is a part of the outside loop. Repeating loops, and also bypass loops, may be set up using a "go to" statement. In a repeating loop the "go to" statement directs the computer to a point in the program where a sequence of steps is repeated. In a by-pass loop the computer is instructed to by-pass certain steps and "go to" a certain point.

The print statement in a program directs the computer to print out a message or quantity. If asked to print X, the value of X is printed and not the letter "X". To print a message, symbols, or a letter as such, enclose them in quotation marks. Following are examples of loops and print statements in excerpts from hypothetical programs.

Example B1:

```
50 FOR I = 1 TO N
60 READ A(I), B(I)
70 NEXT I
```

(A(I) is read:
A sub I)

"I" is used as a subscript, N as the upper limit of the subscript. A(I) and B(I) in the first pass through the loop will read two values from a data line for A(I) and B(I), store it in the computer memory, and then proceed to line 70. It is then directed to the next "I" in line 50, which is now 2. Two values are again read for A(2) and B(2) and stored. This procedure continues until N values of A (to A(N)) and of B have been read. This terminates this loop and the computer proceeds to the next step in the program.

Example B2:

```
20 FOR I = 1 TO N
30 FOR K = 1 TO M
40 LET X(I,K) =
   A(I) * B(K)
50 NEXT K
60 NEXT I
```

Here a loop is inside another loop. As the program progresses through these loops, in the first pass the first I, (I 1), is taken which is a subscript to both A and X. In the next step the first K,(K 1), is taken and these values are used to calculate the first X(I,K), which is X(1,1) A(1)*B(1). The next step directs to the next K,(K 2), in line 30. This time X(1,2) A(1)*B(2) is

(XI,K) is read:
X sub I,K)

calculated. To complete this phase M values of X are calculated using only the A(1) value. The computer then proceeds to the next step which directs to the next I, (I 2), in line 20. With this new value for I, the computer proceeds to the K Loop again and calculates M values of X again, this time using the A(2) value of each of the values for B from B(1) to B(M). This procedure continues until all values of I through N have been used.

Example B3:

```
40 LET I = I + 1
50 GO TO 20
```

Line 50 instructs the computer to go to line 20 and continue the program at that point.

Example B4:

```
70 GO TO 90
80 PRINT "NO SOLUTION"
90 PRINT "FOR X=" X(K)
```

Here line 70 instructs the computer to bypass line 80 and "go to" line 90, where it prints: FOR X=4, (where we assume the value for X(K) to be 4). If executed, line 80 would print: NO SOLUTION

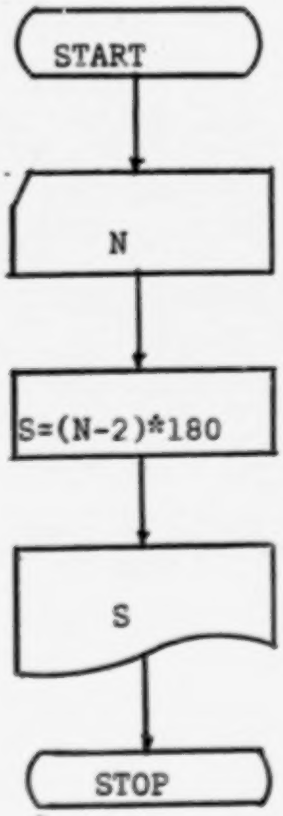
Example B5:

```
70 IF K<N THEN 20
80 PRINT "X("K")=" X(K)
```

Line 70 instructs the computer to compare the K value at this point to the N value and make a decision. If $K \leq N$, then it goes to line 20 and continues the program at that point. If $K > N$, then the computer proceeds to the next step (suppose K = 3 and X(3) = 12.6) and prints: X(3) = 12.6.

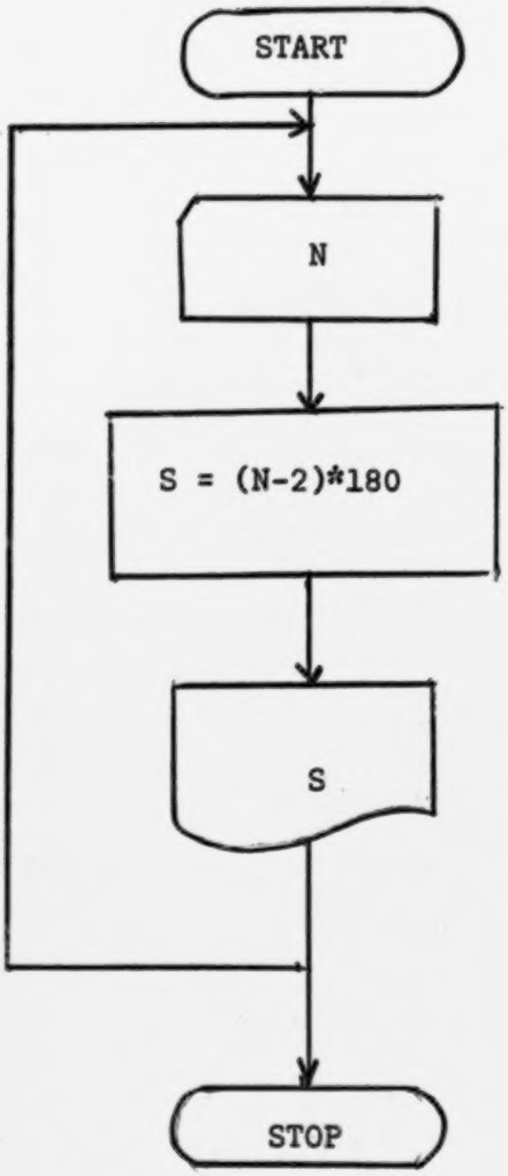
EXAMPLE C

```
ØN AT  15:43   PX MØN 08/21/67  TTY 7
USER NUMBER --P61000
SYSTEM--BAS
NEW ØR ØLD--NEW
NEW PROBLEM NAME--PØLY1
READY.
TAPE
READY.
10 INPUT N
20 LET S=(N-2)*180
30 PRINT S
40 END
RUN
WAIT.
PØLY    15:44   PX MØN 08/21/67
? 8
 1080
TIME:  1 SECS.
RUN
PØLY    15;45   PX MØN 08/21/67
? 12
 1800
TIME:   1 SECS.
RUN
PØLY    15:45   PX MØN 08/21/67
? 24
 3960
TIME:   1 SECS.
BYE
*** ØFF AT 15:46   PX MØN 08/21/67.
```



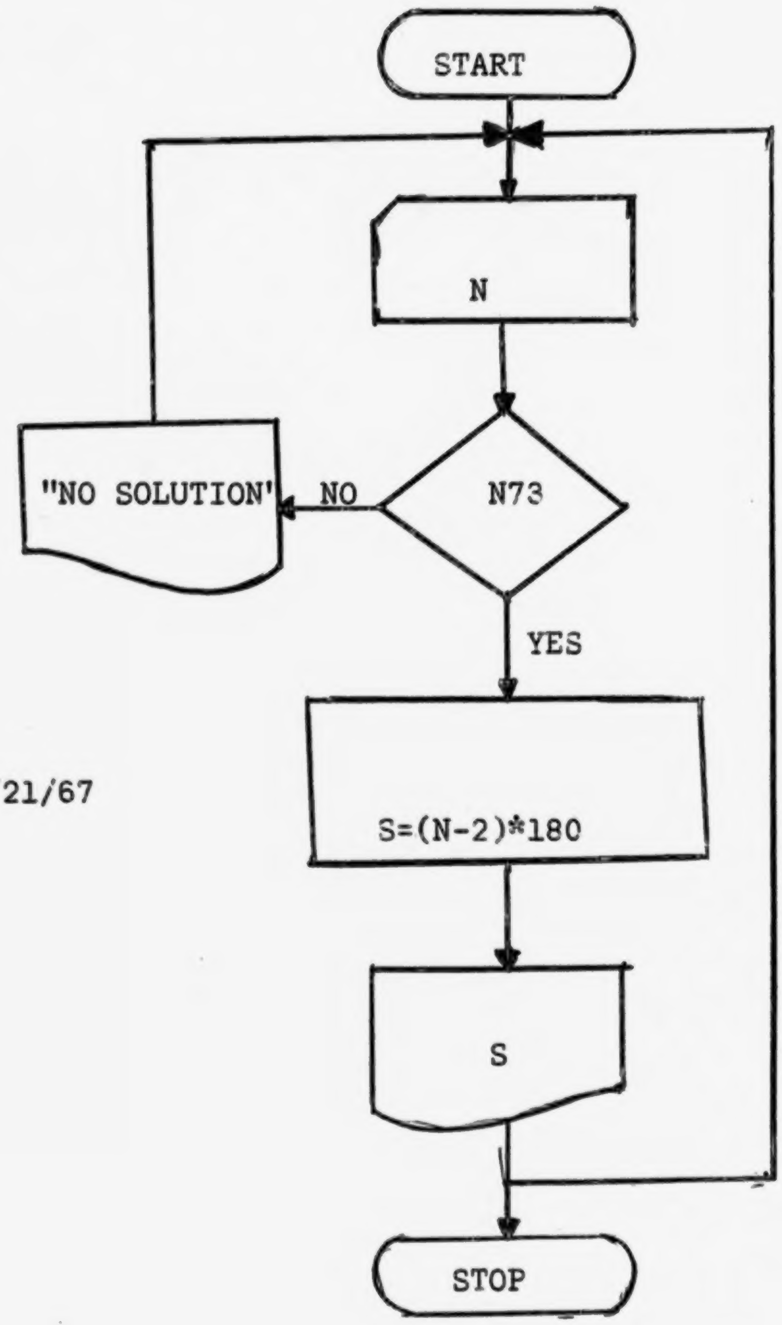
EXAMPLE D

```
ØN AT   15:47   PX MØN 08/21/67    TTY

USER NUMBER--P61000
SYSTEM--BAS
NEW ØR ØLD--NEW
NEW PRØBLEM NAME--PØLY2
READY.

TAPE
READY.

10 INPUT N
20 LET S=(N-2)*180
30 PRINT S
40 GØ TØ 10
50 END
KEY
READY.

RUN


PØLY    15:48   PX MØN 08/21/67

? 8
 1080
? 12
 1800
? 24
 3960
? STØP


RAN    3 SEC.


STØP.
READY.

BYE

*** ØFF AT 15:49   PX MØN 08/21/67.
```



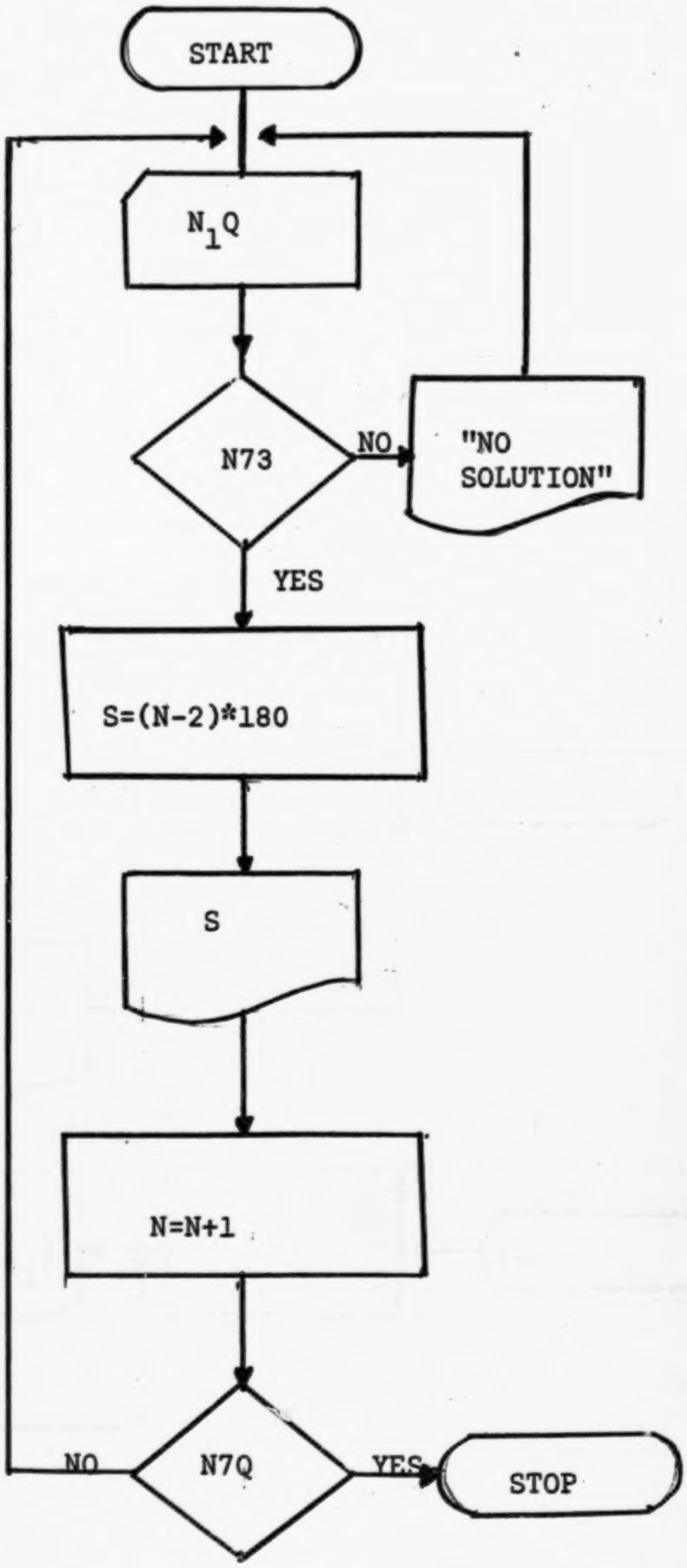
EXAMPLE E

```
ØN AT 15:50  PX MØN 08/21/67

USER NUMBER--P61000
SYSTEM--BAS
NEW ØR ØLD--NEW
NEW PRØBLEM NAME--PØLY3
READY.

TAPE
READY.


10 INPUT N
20 IF N<3 THEN 60
30 LET S=(N-2)*180
40 PRIH←NT S
50 GØ TØ 10
60 PRINT "NØ SØLUTIØN"
70 GØ TØ 10
80 END
KEY
READY.

RUN
WAIT.


PØLY     15:52  PX MØN 08/21/67


? 2
NØ SØLUTIØN
? 3
 180
? 8
 1080
? 12
 1800
? STØP


RAN     5 SEC.



STØP.
READY.
BYE
*** ØFF AT 15:53   PX MØN 08/21/67.
```


START
N
"NO SOLUTION"
NO
N73
YES
S=(N-2)*180
S
STOP

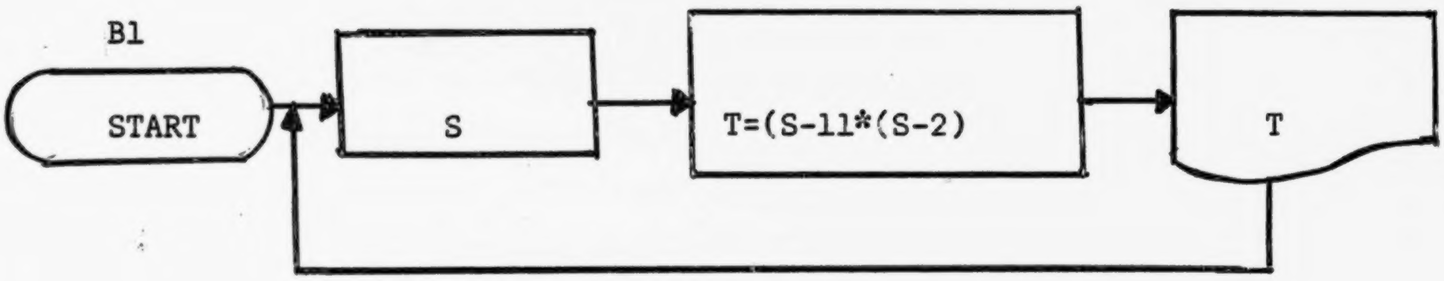
EXAMPLE F

```
ON      15:54    PX MØN 08/21/67   TTY   7
USER NUMBER--P61000
SYSTEM--BAS
NEW ØR ØLD--NEW
NEW PRØBLEM NAME--PØLY4
READY.
TAPE
READY.
10 READ N,Q
20 IF N>=3 THEN 60
30 PRINT "NØ SØLUTIØN"
40 LET N=N+1
50 GØ TØ 20
60 LET S=(N-2)*180
70 PRINT S
80 LET N=N+1
90 IF N<=Q THEN 20
100 DATA 0,25
110 END
KEY
READY.
PØLY    15:56   PX MØN 08/21/67
NØ SØLUTIØN
NØ SØLUTIØN
NØ SØLUTIØN
 180
 360
 540
 720
 900
 1080
 1260
 1440
 1620
 1800
 1980
 2160
 2340
 2520
 2700
 2880
 3060
 3240
 3420
 3600
 3780
 3960
 4140
TIME:  0 SECS.
BYE
** ØFF AT 15:56  PX MØN 08/21/67.
```

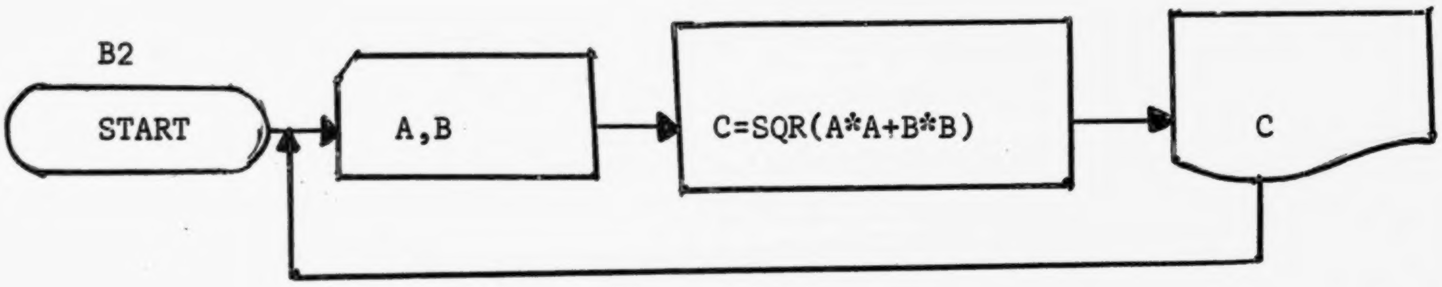

START
N1Q
N73
NO
"NO SOLUTION"
YES
S=(N-2)*180
S
N=N+1
NO
N7Q
YES
STOP

Exercises
Write Basic Programs Using the Following Flow Charts

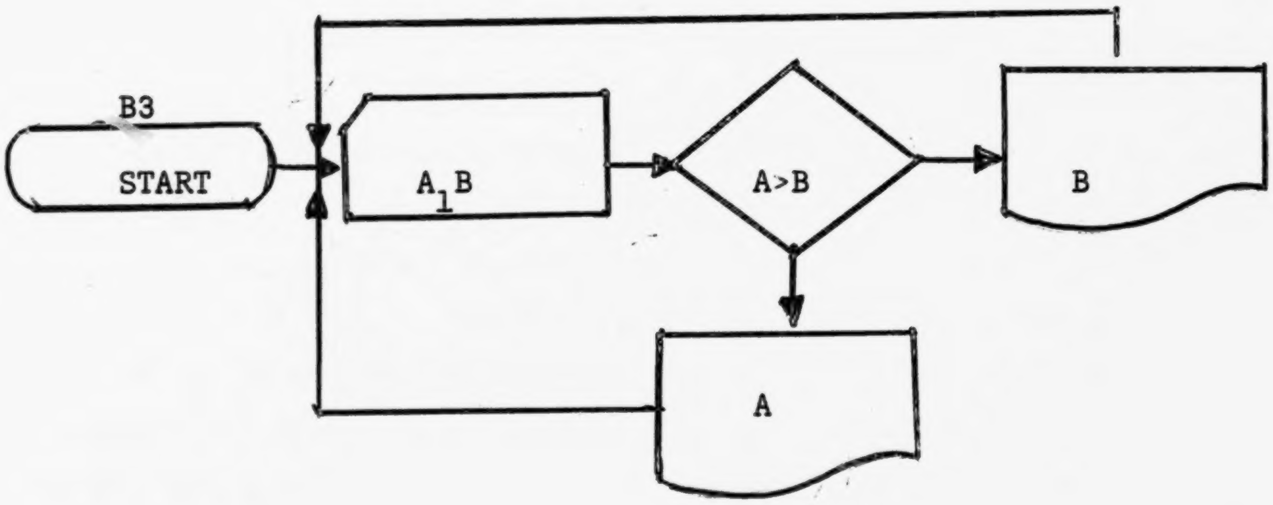
B1


START
S
T=(S-11*(S-2)
T

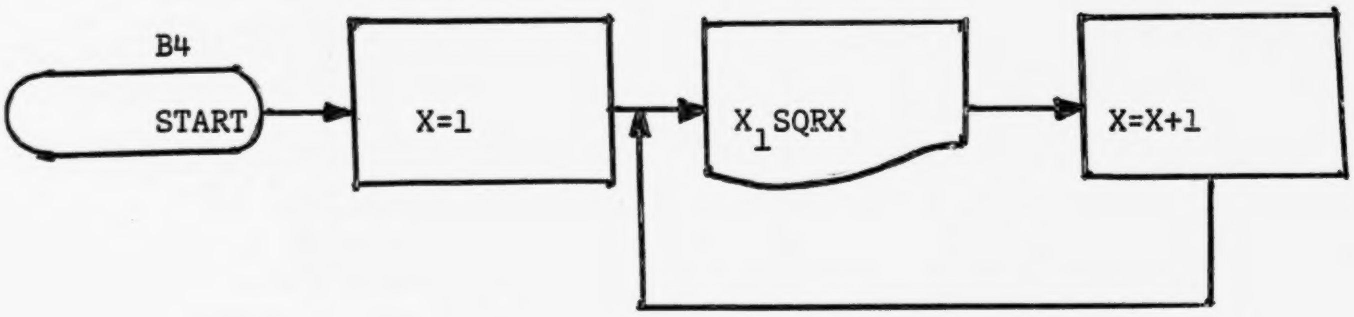
B2


START
A,B
C=SQR(A*A+B*B)
C

B3


START
A,B
A>B
B
A

B4


START
X=1
X,SQRX
X=X+1

Discussion of Approaches to Improve a Program

All four examples were programs for the same problem, finding the sum of the interior angles of a polygon. In example C we had to type "RUN" each time before we could input a new value of N. In this example, we used three polygons. We were on the computer three minutes and used three seconds of actual calculation time.

In example D our only addition to program A is statement 40, which sends the computer back to statement 10 for more input. We note that this time we do not have to type "RUN" each time we wish to input a new value of N. We used the computer two minutes and again used these records of actual calculation time.

In example E we have a more efficient program simply because statement 20 will not allow the operator to use a value of N less than three. In examples D and E notice that the computer continued waiting for data until we typed "STOP".

In example F notice that we have replaced statement ten with the term "READ". What is the purpose of this program? Here we desire to find the sum of the interior angles of each of several polygons beginning with a triangle and increasing the number of sides by one each time. N is the beginning number of sides and Q is the maximum number of sides we wish to use. Again note that if N is less than three we immediately increase N by one in statement 40 and statement 50 sends the computer back to statement 20 with a new value of N. When N is greater than or equal to three we then go to statement 60 and compute the sum of the interior angles for the particular value of N. Notice in statement 70 that we increase N by one. If N is less than or equal to Q we return again to statement 20 and go through the program again finding the sum of the interior angles for our new value of N. This loop will continue until N becomes greater than Q, then the program will be terminated. Notice that we used the computer two minutes and had less than one second of actual calculation time. Yet we found the sum of the interior angles for twenty-six different values of N.

In the foregoing examples and exercises in flow charts and in BASIC programs, observe that a mathematical model or formula is set up initially to represent a problem or a desired quantity to be calculated. Using this formula, or formulas, a procedure (flow chart) is devised to obtain the solution. In simpler problems these formulas are quite obvious. In more complex problems it is sometimes difficult to set up a formula that will generate the desired quantity in a manner that lends itself to computer application. Often a series of terms are involved in a generation. To reserve a separate part of the program for each term desired would be prohibitive in time and space required for writing such a program. Use of loops to generate such quantities is much more efficient but requires a certain amount of insight and skill in mathematical procedures. To develop skill in this area requires much practice and application. Expanding computer application to include more complex geometric problems and those related to sequences are illustrated here in a limited sense to give an insight into the possibilities that exist, or may exist through future research, in the field of computer applications. In-depth study may be pursued as the teacher or student chooses. On completion of this elementary study presented here, the student is encouraged to continue in individual investigations and studies. Following are examples illustrating applications in geometry, sequences, and even in recreation.

```
CIR      16:05   PX TUE 08/22/67
TANGENT EXTERNALLY             3         2              5
 9              4              5        TANGENT INTERNALLY
 2              2              5        EXTERNAL NØNINTERSECTING
 10             7              2.23607  INTERNAL NØNINTERSECTING
 10             8              2.23607  INTERSECT
CIRCLES ARE CØNCENTRIC
CIRCLES ARE THE SAME
ØUT ØF DATA IN  10
TIME:    1 SECS.
LISTNH
10 READ R1,R2,X1,Y1,X2,Y2
20 IF R1<R2 THEN 160
30 LET D=SQR((X1-X2)↑2+(Y1-Y2)↑2)
40 IF D=0 THEN 110
50 IF R1+R2=D THEN200
60 IF R1-R2=D THEN 220
70 IF R1+R2<D THEN 240
80 IF R1-R2>D THEN 260
90 IF R1-R2<D THEN 280
100 GØ TØ 10
110 IF R1=R2 THEN 140
120 PRINT "CIRCLES ARE CØNCENTRIC"
130 GØ TØ 10
140 PRINT "CIRCLES ARE THE SAME"
150 GØ TØ 10
160 LET T=R1
170 LET R1=R2
180 LET R2=T
190 GØ TØ 50
200 PRINT "TANGENT EXTERNALLY",R1,R2,D
210 GØ TØ 10
220 PRINT R1,R2,D,"TANGENT INTERNALLY"
230 GØ TØ 10
240 PRINT R1,R2,D,"EXTERNAL NØNINTERSECTING"
250 GØ TØ 10
260 PRINT R1,R2,D,"INTERNAL NØNINTERSECTING"
265 GØ TØ 10
```

```
280 PRINT R1,R2,D,"INTERSECT"
290 GØ TØ 10
300 DATA 3,2,2,3,-1,-1,9,4,2,3,-1,-1,2,2,2,2
305 DATA -1,-2,10,7,1,1,-1,0,10,8,1,1
310 DATA -1,0,6,5,5,4,5,4,6,6,5,4,5,4
670 END
```

```
NEW
NEW PRØBLEM NAME--SUM100
READY.
TAPE
READY.
10 PRINT"SUM ØF THE FIRST 100 INTEGERS"
20 READ N
30 LET K=1
40 LET S=0
50 LETS=S+K
60 LET K=K+1
70 IF K<=N THEN 50
80 PRINT"S="S
90 DATA 100
100 END
KEY
READY.
RUN
SUM 100    8:42    PX THU 08/24/67
SUM ØF THE FIRST 100 INTEGERS
S= 5050
TIME:  0 SECS.
BYE
*** ØFF AT 8:42   PX THU 08/24/67.
```

```
SQRRTS    8:39   PX THU 08/24/67
SEQUENCE ØF SQUARE RØØTS
X              SQR[X]
 1             1
 2             1.41421
 3             1.73205
 4             2
 5             2.23607
 6             2.44949
 7             2.64575
 8             2.82843
 9             3
 10            3.16228
 11            3.31662
 12            3.4641
 13            3.60555
 14            3.74166
 15            3.87298
 16            4
 17            4.12311
 18            4.24264
 19            4.3589
 20            4.47214
 21            4.58258
 22            4.69042
 23            4.79583
 24            4.89898
 25            5
TIME:  0 SECS.
LISTNH
5 PRINT"SEQUENCE ØF SQUARE RØØTS"
10 READ N
20 PRINT"X","SQR(X)"
30 FØR X=1 TØ N
40 PRINT X,SQR(X)
50 NEXT X
60 DATA 25
70 END
```

NOTE: LISTNH and the program printed below the solutions serve as a means of making corrections in the original program and having the computer type out a corrected program. LISTNH is typed to instruct the computer to accomplish this.

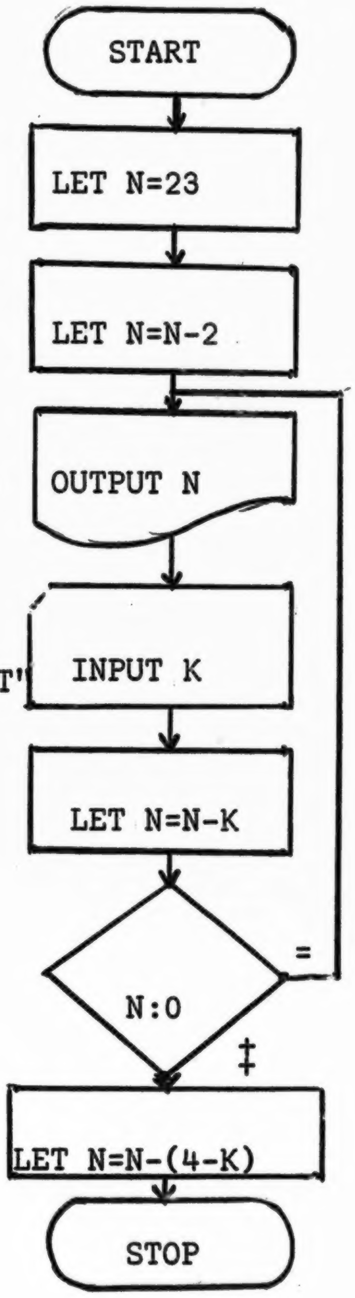
Installments where Monthly Payments Pay Interest Due for the Previous Month and the Balance of the Payment is Applied to the Principal: $1000@6%

```
PAYM        10:08     PX THU 08/24/67
5 PRINT"NØ.","PAYMENT","INTEREST","ØN PRINCIPAL","BALANCE ØN P"
10 READ R,D,A
20 LET K=1
30 LET J=1
40 LET B=R*D/12
50 LET C=A-B
60 LET D=D-C
70 PRINT K,A,B,C,D
80 LET K=K+1
90 IF J>=2 THEN 160
100 IF D>=A+B THEN 40
110 LET C=D
120 LET A=C+B
130 LET J=J+1
135 LET D=0
140 GØ TØ 70
150 DATA .06, 1000,60
160 END
RUN
PAYM       10:08    PX THU 08/24/67
NØ.            PAYMENT          INTEREST         ØN PRINCIPAL     BALANCE ØN P
 1              60               5.               55               945
 2              60               4.725            55.275           889.725
 3              60               4.44863          55.5514          834.174
 4              60               4.17087          55.8291          778.344
 5              60               3.89172          56.1083          722.236
 6              60               3.61118          56.3888          665.847
 7              60               3.32924          56.6708          609.177
 8              60               3.04588          56.9541          552.223
 9              60               2.76111          57.2389          494.984
 10             60               2.47492          57.5251          437.459
 11             60               2.18729          57.8127          379.646
 12             60               1.89823          58.1018          321.544
 13             60               1.60772          58.3923          263.152
 14             60               1.31576          58.6842          204.468
 15             60               1.02234          58.9777          145.49
 16             60               .727449          59.2726          86.2173
 17             60               .431087          59.5689          26.6484
 18             27.0795          .431087          26.6484          0
 TIME:    1 SECS.
```

Game of 23 Matches

In the 23 match game there are two players. Each in turn must pick up 1, 2, or 3 matches. The player who is forced to pick up the last match loses.

```
23GAME     12:56     PX WED 08/23/67
10 PRINT"23 MATCH GAME:KØM PEWTER VS.YØU"
20 LET N=23
30 LET N=N-2
40 PRINT"I,KØM PEWTER,TAKE 2 MATCHES"
50 PRINT"THERE ARE"N"LEFT.HØW MANY DØ YØU TAKE?"
60 LET I=1
70 INPUT K
80 IF 4-K>3 THEN 160
90 IF 4-K<1 THEN 160
100 LET N=N-K
110 IF N=0 THEN 240
120 LET N=N-(4-K)
130 PRINT"I,KØMPEWTER,TAKE"4-K"MATCHES."
140 PRINT"THERE ARE"N"MATCHES LEFT.HØW MANY DØ YØU TAKE?"
150 GØ TØ 70
160 LET I=I+1
170 IF I<3 THEN 212
180 IF I>3 THEN 220
190 PRINT"THAT'S TWICE YØU'VE CHEATED-ØNCE MØRE AND I QUIT"
200 PRINT"NØW, CHØØSE EITHER 1,2,ØR 3 MATCHES"
210 GØ TØ 70
212 PRINT"YØU CHEATED.CHØØSE ØNLY 1,2,ØR 3 MATCHES"
215 GØ TØ 70
220 PRINT"I QUIT"
230 GØ TØ 250
240 PRINT"HØW ABØUT THAT. I WØN AGAIN.WHØ'S NEXT?"
250 END
RUN
23GAME    12:59    PX WED 08/23/67
23 MATCH GAME:KØM PEWTER VS.YØU
I,KØM PEWTER,TAKE 2 MATCHES
THERE ARE 21   LEFT.HØW MANY DØ YØU TAKE?
? 1
I,KØMPEWTER,TAKE 3   MATCHES.
THERE ARE 17   MATCHES LEFT.HØW MANY DØ YØU TAKE?
? 3
I,KØMPEWTER,TAKE 1   MATCHES.
THERE ARE 13   MATCHES LEFT.HØW MANY DØ YØU TAKE?
? 5
YØU CHEATED.CHØØSE ØNLY 1,2,ØR 3 MATHCES
? 2
I,KØMPEWTER,TAKE 2   MATCHES.
THERE ARE 9    MATCHES LEFT.HØW MANY DØ YØU TAKE?
? .5
THAT'S TWICE YØU'VE CHEATED-ØNCE MØRE AND I QUIT
NØW, CHØØSE EITHER 1,2,ØR 3 MATCHES
? 2
I,KØMPEWTER,TAKE 2   MATCHES.
THERE ARE 5    MATCHES LEFT.HØW MANY DØ YØU TAKE?
? 3
I,KØMPEWTER,TAKE 1   MATCHES.
THERE ARE 1    MATCHES LEFT.HØW MANY DØ YØU TAKE?
? 1
HØW ABØUT THAT. I WØN AGAIN.WHØ'S NEXT?
TIME:  13 SECS.
```



THE INTERSECTION OF CIRCLES

by Nick R. Kalianov & Pearl Price

Our purpose is to investigate, using analytic methods, the problem of the intersection of two circles lying in the same plane.

Consider two circles with centers $A(x_1,y_1)$ and $B(x_2,y_2)$ and radii r_1 and r_2 respectively. Let $d = m(\overline{AB})$. Assume $r_1 \geq r_2$.

In Figure 1, a coordinate system is set up so that A is the origin, $\overleftrightarrow{AB}$ is the x-axis; and the x-coordinate of B is nonnegative. The equations of the circles are: (1) $x^2 + y^2 = r_1^2$ and (2) $(x-d)^2 + y^2 = r_2^2$.

For a point P to belong to both circles, the coordinates of P must satisfy both (1) and (2). If we subtract (2) from (1), we get (3) $(x^2 + y^2) - [(x - d)^2 + y^2] = r_1^2 - r_2^2$. Hence, a point P belongs to both circles if and only if its coordinates satisfy both (1) and (3). Equation (3) can be simplified algebraically to (3') $2dx = r_1^2 - r_2^2 + d^2$. Show this simplification.

We can conclude from (3') that if $d = 0$ there are points common to both circles only if $r_1 = r_2$, i.e., the circles are the same. Also, if $d = 0$ and $r_1 \neq r_2$, then the circles do not intersect.

Let's assume that $d \neq o$. Then P belongs to both circles if and only if (1) $x^2 + y^2 = r_1^2$ and (2') $x = \frac{r_1^2 - r_2^2 + d^2}{2d}$. Note that (2') is the equation of a line perpendicular to the x-axis. Since $r_1 \geq r_2$, $\frac{r_1^2 - r_2^2 + d^2}{2d} \geq 0$. Hence we can reduce the problem to the intersection of a circle (with the radius r_1) and a line (where the distance between the line and the center of the circle is $\frac{r_1^2 - r_2^2 + d^2}{2d}$.).

We may now conclude that, if $d \neq 0$, the two circles intersect in two points if and only if $\frac{r_1^2 - r_2^2 + d^2}{2d} < r_1$. Simplifying the inequality, we find that the two circles intersect in two points if and only if $d < r_1 + r_2$ and $d > r_1 - r_2$.

The two circles will intersect in one point if and only if $\frac{r_1^2 - r_2^2 + d^2}{2d} = r_1$. Simplifying the equality, we find that the two circles will intersect in one point if and only if $d = r_1 - r_2$ or $d = r_1 + r_2$.

The intersection of the two circles will be the empty set if and only if $\frac{r_1^2 - r_2^2 + d^2}{2d} > r_1$. Simplifying the inequality, we find that the two circles do not intersect if and only if $d < r_1 - r_2$ or $d > r_1 + r_2$.

We have now completed our investigation. See Figure 2 for a summary of our analysis.

Figure 3 is a flow chart that graphically depicts an overall view of the above procedure. The flow chart must be logically complete and must lead step by step to the desired answer. Do you see that it does?

As an exercise for the student, select four pairs of circles, noting the coordinates of the center and the radius of each circle. For each pair of circles, follow the flow chart through until you reach a print statement. Check the result of the flow chart graphically.

We have written out the program in BASIC directly from the flow chart. Study this program, using the flow chart as a guide, to see that it is correct. The program includes data for five pairs of circles that are externally tangent, are the same, intersect in two points, do not intersect, and do not intersect, respectively.

As an exercise for the student, run this program with data statements of your own.

Exercises

1. In our discussion of the intersections of circles, we made use of previous knowledge concerning the intersection of a circle and a line. Construct a flow chart, and write a program that will determine whether or not:

   a. the circle and line intersect in two points,
   b. the circle and line intersect in one point
   c. the circle and line do not intersect.

2. Using the program above, describe the relative position of two circles of radii 5 and 9 if the center line has the length (a) 14 (b) 4 (c) 12 (d) 16.

PROGRAM

```
10   Read x1, y1, x2, y2, r1, r2
20   If r1> = r2 then 50
30   Print " Pick r1> = r2"
40   Go to 10
50   Let D = SQR( (x2 - x1)^2 + (y2 - y1)^2 )
60   If x1 = x2 then 120
70   If D = r1 + r2 then 190
```

```
80    If D = r1 - r2 then 210
90    If D > r1 + r2 then 230
100   If D > r1 - r2 then 250
110   Gø tø 150
120   If y1 = y2 then 140
130   Gø tø 70
140   If r1 = r2 then 170
150   Print "Circles are Concentric"
160   Gø tø 10
170   Print "The circles are the same"
180   Gø tø 10
190   Print "The circles are externally tangent"
200   Gø tø 10
210   Print "The circles are internally tangent"
220   Gø tø 10
230   Print "One circle lies outside the other"
240   Gø tø 10
250   Print "The circles intersect in two points"
260   Gø tø 10
270   Data 0,0,0,2,1,1,0,0,0,0,4,4
280   Data 2,0,0,2,3,2,0,0,0,1,4,1
290   Data 0,0,5,5,2,1
300   End
```

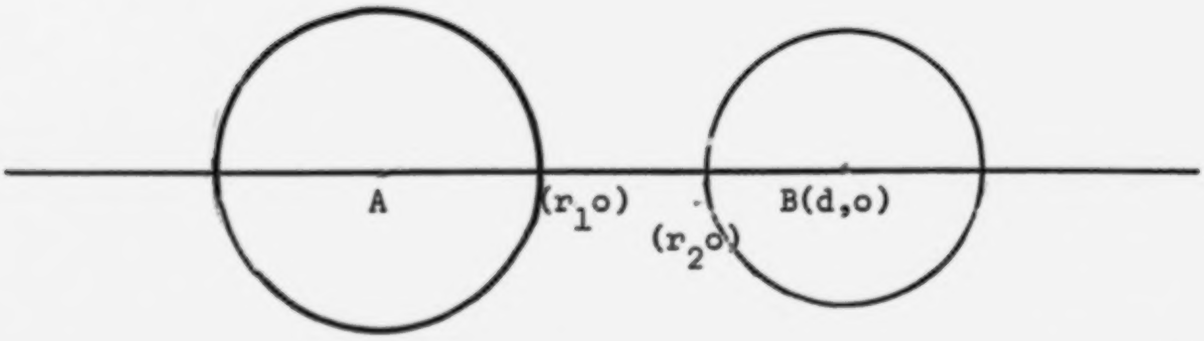
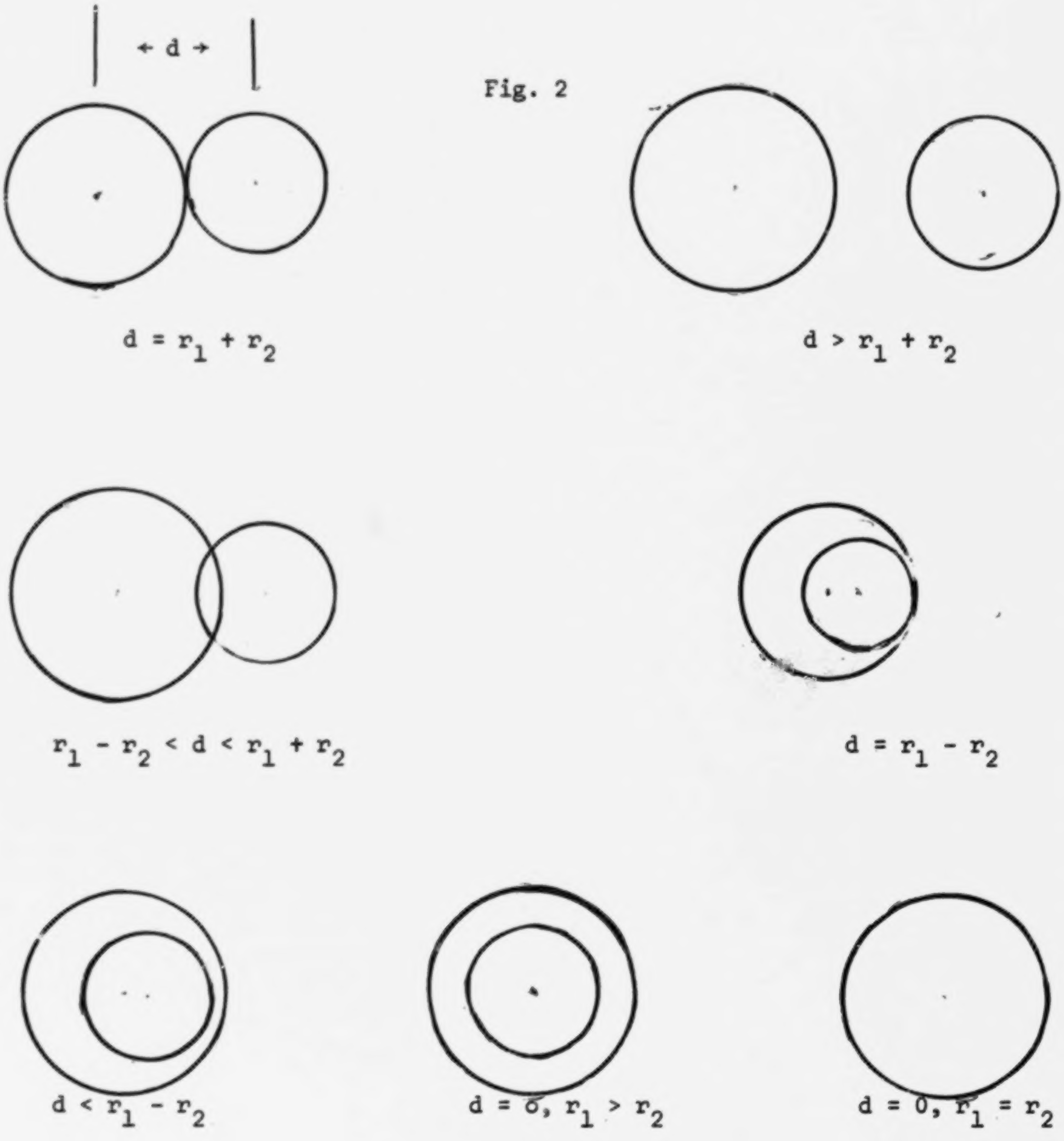

A
$(r_1 o)$
$(r_2 o)$
$B(d,o)$

Fig. 1

Fig. 2



$d = r_1 + r_2$

$d > r_1 + r_2$

$r_1 - r_2 < d < r_1 + r_2$

$d = r_1 - r_2$

$d < r_1 - r_2$

$d = 0,\ r_1 > r_2$

$d = 0,\ r_1 = r_2$

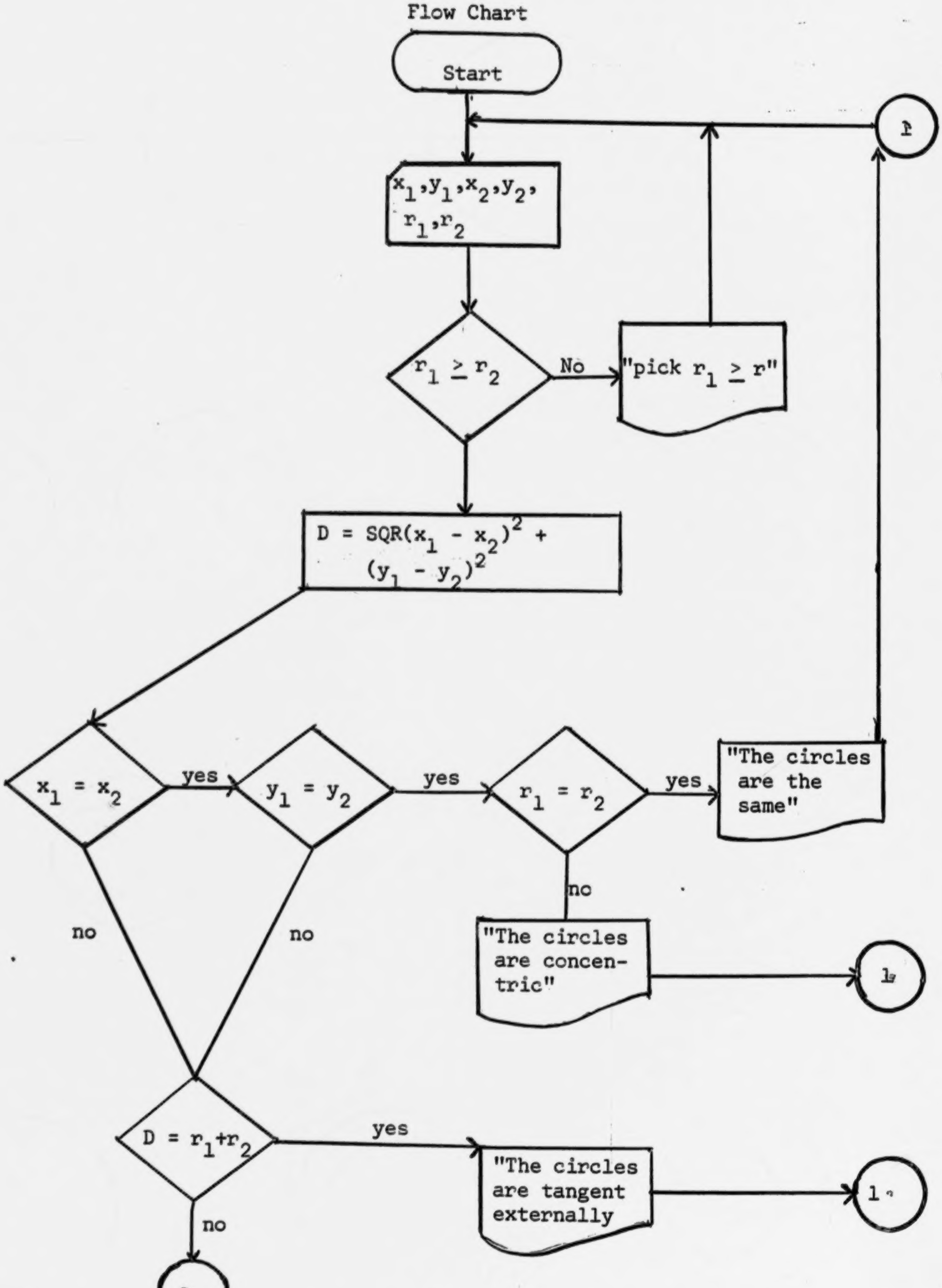

Flow Chart
Start
1
x1,y1,x2,y2,
r1,r2
r1 ≥ r2
No
"pick r1 ≥ r"
D = SQR(x1 - x2)² +
(y1 - y2)²
x1 = x2
yes
y1 = y2
yes
r1 = r2
yes
"The circles are the same"
no
no
no
"The circles are concentric"
1
D = r1+r2
yes
"The circles are tangent externally
1
no
2

Fig. 3

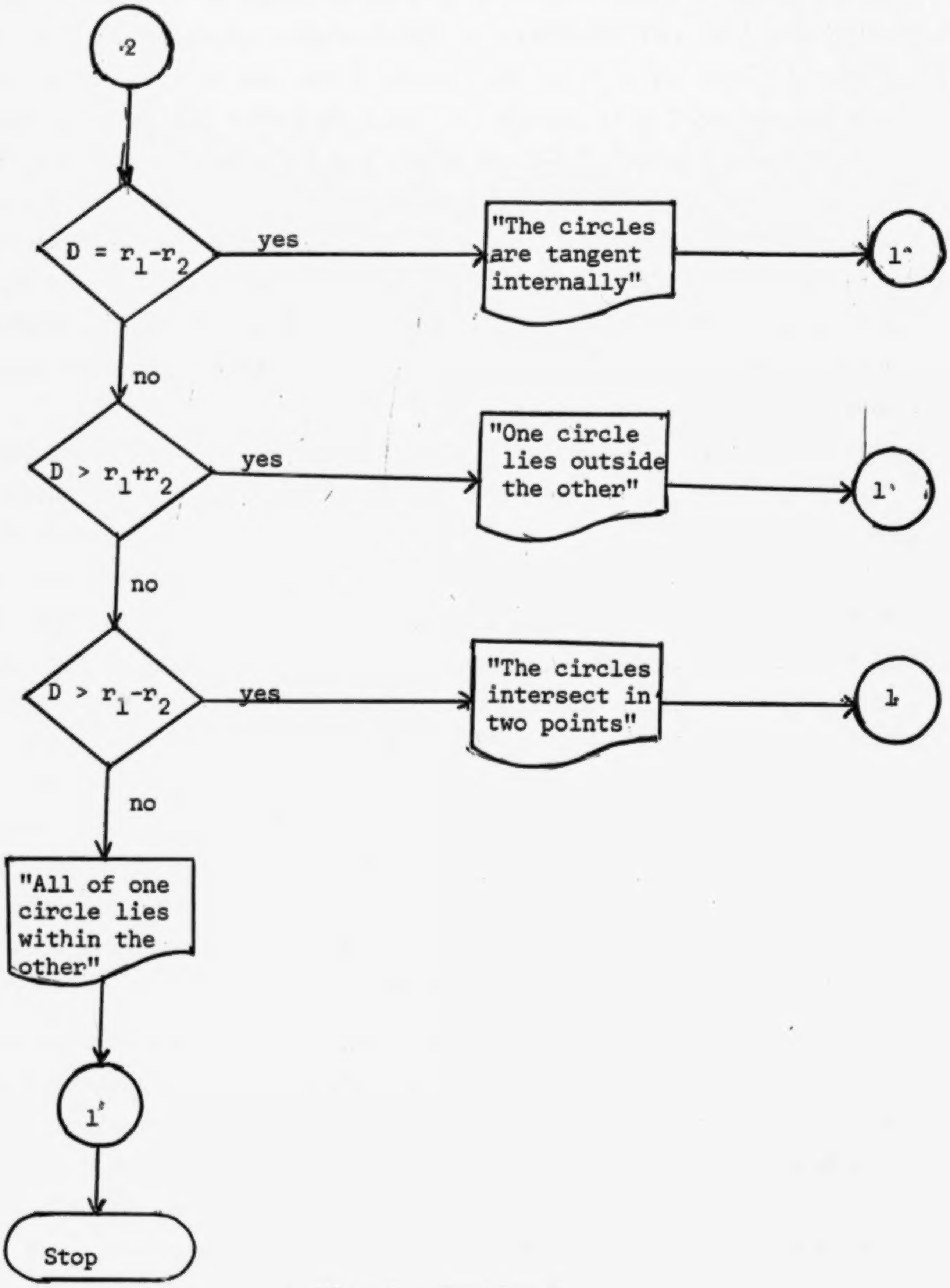

2
D = r_1-r_2
yes
"The circles are tangent internally"
1
no
D > r_1+r_2
yes
"One circle lies outside the other"
1
no
D > r_1-r_2
yes
"The circles intersect in two points"
1
no
"All of one circle lies within the other"
1
Stop

Fig. 3 - Continued

FEEDBACK AROUND AN INTEGRATOR

by Dean C. Larsen

This unit presents an additional section for use with the "Feedback" chapter (C-1) in the Engineering Concepts Curriculum Project text. The unit contains a presentation of a computer model of feedback at a level which would be understandable to the average, college bound high school student. There is a short discussion of why the material is included in the text and how it is presented. Following this discussion, there is a detailed description of how one might build a model of a dynamic feedback system in the computer. It is hoped that this model will help students understand the concept of feedback. It is assumed that each student has some exposure to BASIC computer language and that section B of the text has been covered. Section B deals with the model of dynamic systems, and it is assumed that students recognize the significance of the area under a rate of change curve.

During the last ten years, engineers have begun to recognize feedback as the underlying principle behind any system in which there is goal seeking behavior. It is this universal applicability of the feedback concept to both physical and nonphysical systems which requires its presentation in this text. The essence of feedback is that the output of a system is used to modify the input into the system so that the output is changed to conform to some predetermined goal. Many qualitative examples of feedback are introduced at the first of the chapter to help students develop their understanding of this simple but elusive concept.

In thinking about feedback, consider the system to be a student and all elements relating to his progress in a difficult course. The output of the system is the set of grades which the student earns. The input is all the study and work which the student does. How does the concept of feedback relate to this system?

Assume that the goal for the system is a cumulative grade average of A in the course for this student. During the first week of school the student goes to class, takes a few notes and in general tries to understand the material. At the end of the week the teacher gives a quiz. The student gets some grade. This is the first output. Now feedback begins to take over and should direct the system toward the goal. The student makes a comparison between the grade (output) and the kind of study he is doing (input). This comparison causes the student to change his input in a direction which will force the output toward the desired goal. If the quiz grade is low, the feedback causes the student to adjust his rate of study (input) so as to bring the level of output up. If the grade (output) is higher, less correction in the rate of study is needed. Note that in this qualitative example of feedback, output changes input to produce changed output. This is feedback.

Before leaving this example, note how the quality of the feedback control would be improved if the student were aware of the "SUM" of all of his grades in the course as opposed to just being aware of the "last" grade earned. Consider the student's situation if all grades were F except the last one, which was A. If the feedback does not keep track of the "SUM" of all grades earned, the student might reduce his rate of study on the basis of the last A and yet be on the verge of failing the class and missing the goal. Remember this idea of "SUM." The student would be better off keeping track of the sum of his grades rather than just his most recent grade.

Many examples of feedback producing goal seeking behavior are used in the text. Feedback in physical systems is discussed, and students should have a feel at this point for the presence of feedback in a machine -- directing the system toward a goal. Following this treatment, several quantitative examples of feedback around a scalor or multiplier are presented. The following example shows the nature of this quantitative description.

Consider the system WITHOUT feedback (Figure 1). The desired output is 5/2 times the input at the SAME INSTANT. The output is nothing but a scaled up picture of the input at the same instant. It is obvious that in this system, output is very sensitive to changes in system characteristics. For example, if the value of the scalor A is reduced by 50% to 5/4, the output also drops by 50%.

Consider the system WITH feedback (Figure 2). The output of this system will be the same as the output of the previous system. The feedback loop will cause this system to be less sensitive to changes in system characteristics. It is important to realize that in this system, the input at any instant is a function of the output at that same instant. That is, output modifies input at the same instant. Hence, we can say:

$$y = (x - 0.4y)(A)$$

This means that the output y is a function of the input x which is modified by a factor of the output y at the same instant. Both of the y's in our equation represent output at the same instant, so we can solve the equation for y.

$$y = (x - 0.4y)(A)$$
$$y + 0.4yA = xA$$
$$y(1 + 0.4A) = xA$$
$$y\left(\frac{10 + 4A}{10}\right) = xA$$
$$y = \left(\frac{10}{10 + 4A}\right)(xA)$$

Multiply numerator and denominator of the fraction on the right by 1/A.

$$y = \left(\frac{10x}{\frac{10}{A} + 4}\right)$$

Since the output of this system is to be the same as the output of the first system, A is made very large. Then

$$y \cong (5/2)x$$

because $\frac{10}{A}$ is very small.

The text spends considerable time discussing examples such as this. The fact is stressed that the output is relatively independent of large changes in the scalor A.

Let A = 2000 then $y = \left(\frac{10x}{\frac{10}{2000} + 4}\right)$ $y = \frac{10x}{4.005}$ $y \cong (5/2)x$

Let A = 1000 then $y = \left(\frac{10x}{\frac{10}{1000} + 4}\right)$ $y = \frac{10x}{4.01}$ $y \cong (5/2)x$

A 50% drop in the value of the multiplier A causes very little change in the output y.

The preceding discussion is to inform the reader in a general way about the method of presentation in the chapter. There is a problem with this presentation in that most feedback controlled systems are not simple scalors. The output is not a simple multiple of the input at the same instant. The fact is that most controlled systems act more like integrators than scalors. The following material is written to give students insight into the nature of feedback around an integrator. It also provides an interesting example of computer modeling. The material is written as it might appear in the text.

The careful reader will observe a difference between the qualitative examples of feedback (the car on page C-1.2, the ship on page C-1.9) and the latter quantitative examples starting on page C-1.14. The systems which are treated with algebra involve feedback around a scalor. Before reading further, see if you can determine how these systems differ from the qualitative examples at the first of the chapter.

The truth of the matter is that many of the descriptive examples are not true scalors. These systems produce an output which is not a simple scaled up replica of the input. What one must be aware of at this point is that in many real systems, the output is determined by all previous values of the input and not by the most recent value. Consider the following as an example of this idea.

Suppose elevator number 13 is dropping at full velocity past the 25th floor of the "Mile Hi" building at this very instant. The operator is inputing a velocity signal with the control handle, and the elevator is producing a change in location. The operator senses how far the elevator has moved by viewing the floor numbers as they pass by a little window.

25-24-23-22-21-20-19-18-17-16-

The operator remembers the system goal.

"Stop this car at the ground floor after traveling down 25 stories." Quickly the operator makes a comparison between output (9 stories gone) and the input signal. This input signal is really the position of the control handle. This "feedback comparison" causes the operator to change the control handle position (input). The elevator slows but does not stop. It continues to drop at a reduced velocity.

---15---14---13---12---11---10---9---8---7---6

Again the operator senses the output (19 stories gone). A quick comparison of this output with the input (control handle position) again causes him to change the input. The elevator continues to drop but at an even lower velocity.

-----5------4------3------2---

Eventually this feedback procedure will produce the sought after goal of 25 stories displacement in a downward direction. Notice that the output of the elevator (distance traveled) is a result of all previous input velocity signals prior to the time the elevator stops. This elevator is really an integrator.

The output of an integrator at any instant is determined by all previous values of the input up to that instant. Since most of the systems we control in the real world are dynamic, that is they change with time, we can see that they are really integrators and not simple scalors. Let us now develop a systematic example of feedback around an integrator.

V = velocity in ft./sec. -- this is the input signal.
D = displacement in ft. -- this is the output signal.
F = feedback signal
E = corrected velocity signal

Figure 3 represents feedback around an integrator. The variables are defined above. Let the system represent an automobile. The integrator will total or "SUM" up the distance traveled by the automobile after some arbitrary starting time. The feedback loop senses the total distance traveled by the car and corrects or adjusts the velocity accordingly. This feedback signal indicates how much the original velocity should have decreased at the end of any time interval. As the total distance D increases, the feedback loop reduces the velocity to a lower and lower value. The scalor constant -4/10 was picked because of the behavioral goal which we have chosen for the system. The method used to pick this constant is beyond the scope of this book. However, methods are available for picking constants in light of chosen goals. Given an initial velocity of 88 ft./sec., the goal for this system is to slow the car so that the velocity is zero when the car has traveled 220 feet. Assume the following values for the variables at time T = 0 sec.

V = 88 ft./sec. -- Input
T = 0 sec. -- Time
D = 0 ft. The car has not yet gone any distance.

Now, from Figure 3, we can see:

F = -0.4(D)
F = 0, no feedback signal
E = V + F
E = 88 ft./sec. This is the corrected velocity signal.

Now, assume that E holds constant for one second (see Figure 4). At the end of one second, what is the value of D? D is represented by the area under the corrected velocity curve (see Figure 5).

D = 1 sec. x 88 ft./sec.
D = 88 ft.

During the first second the car traveled 88 ft. The new values of the variables after one second are:

T = 1 sec.
D = 88 ft.
F = -0.4 x 88
F = -35.2 ft./sec.

Now, let us see what happens if E remains constant at this new value for one second (see Figure 6). You notice that the total area under the curve has increased during the time interval T = 1 sec. to T = 2 sec. However, the increase in area during this time interval is less than during the time interval T = 0 sec. to T = 1 sec. This is because the car travels at a lower corrected velocity of 52.8 ft./sec. during the interval T = 1 sec. to T = 2 sec.

Increase in area under the curve = 1 sec. x 52.8 ft./sec.
= 52.8 ft.

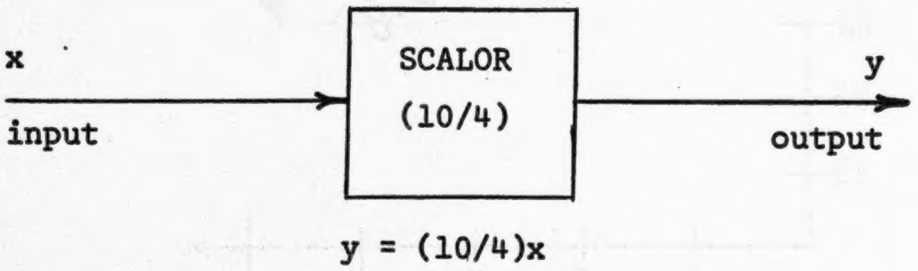

x
input
SCALOR
(10/4)
y
output
y = (10/4)x

Figure 1. Without Feedback

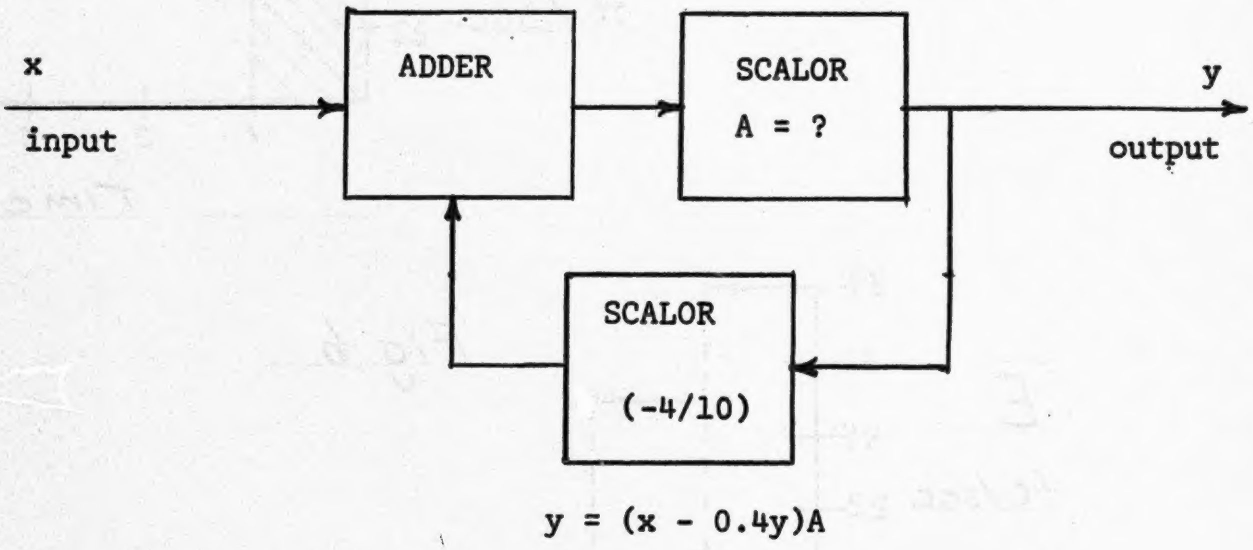

x
input
ADDER
SCALOR
A = ?
y
output
SCALOR
(-4/10)
y = (x - 0.4y)A

Figure 2. With Feedback

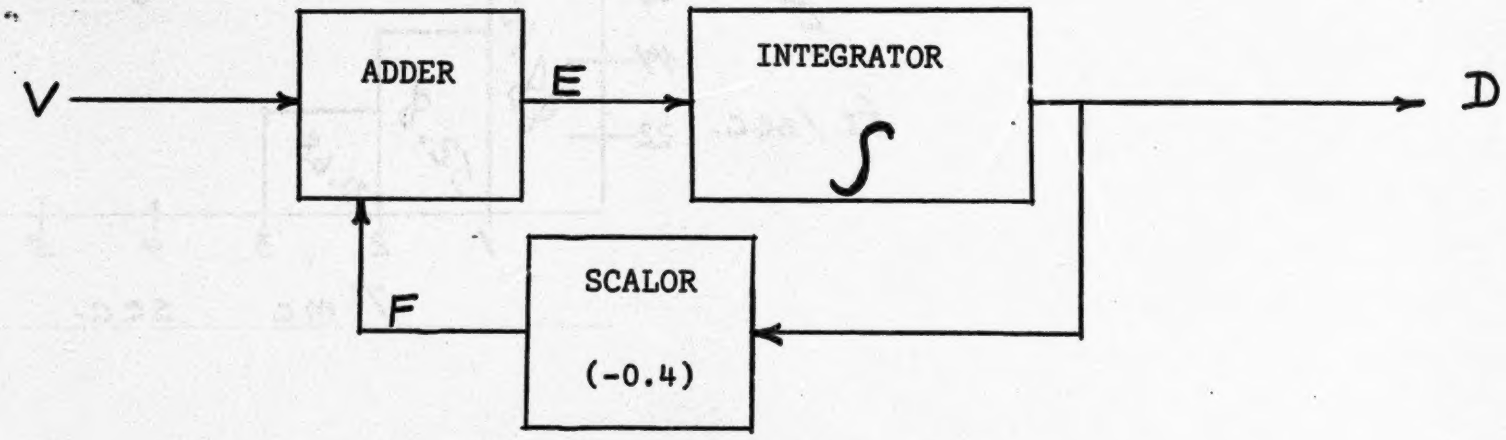

V
ADDER
E
INTEGRATOR
∫
D
F
SCALOR
(-0.4)

Figure 3. Integrator with Feedback

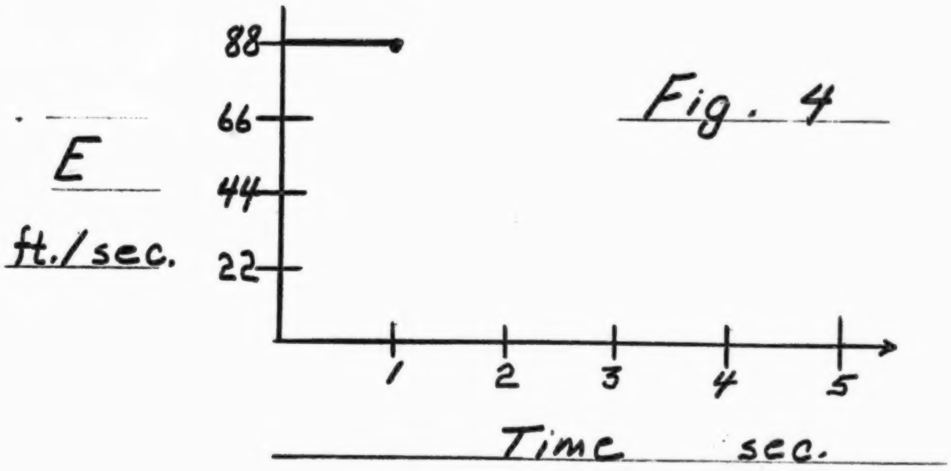


Fig. 4

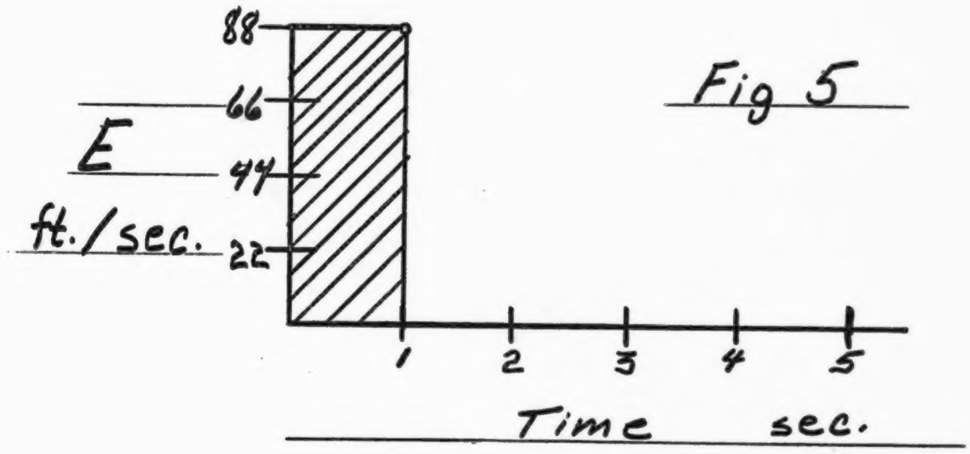


Fig 5

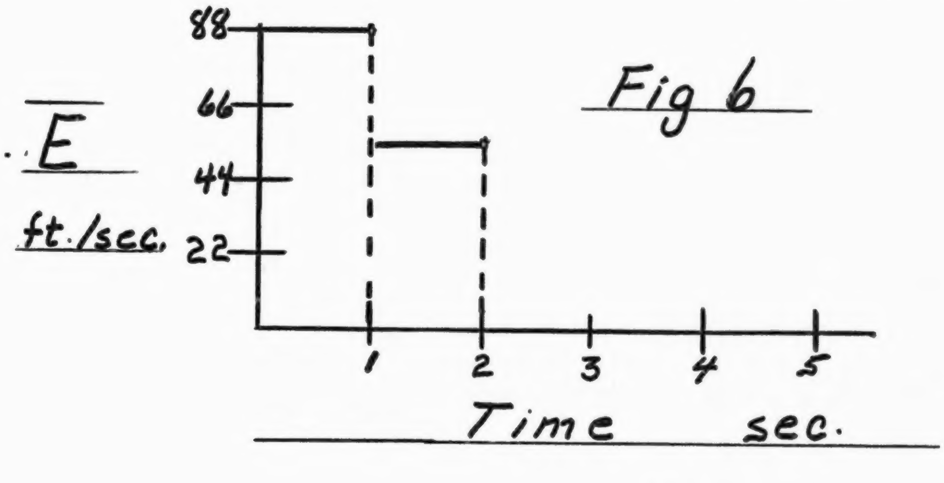


Fig 6

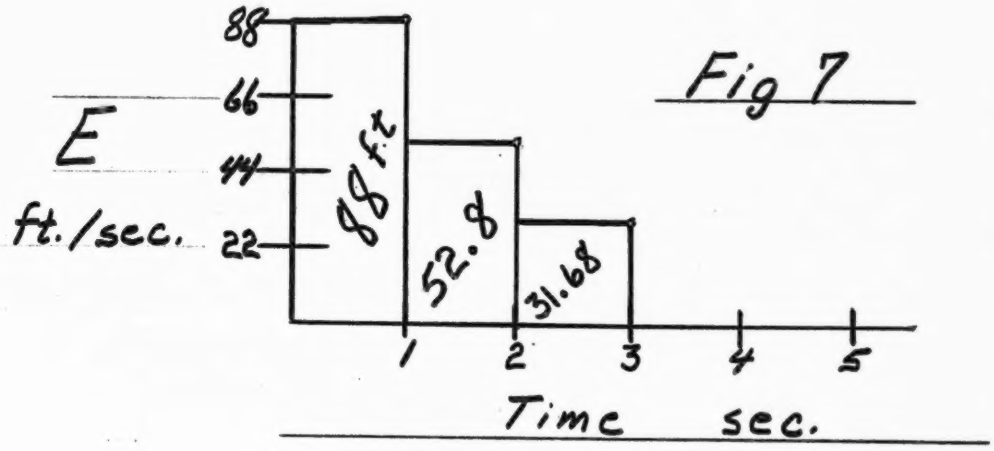


Fig 7

Increase in distance is then = 52.8 ft.
Total Distance = 88 ft. + 52.8 ft.
= 140.8 ft.

The new values of the variables after 2 seconds are:

T = 2 sec.
D = 140.8 ft.
F = -0.4 x 140.8
F = -56.32 ft./sec.

This feedback signal is added to the original velocity signal to get a new value of E, the corrected velocity signal.

E = V + F
E = 88 - 56.32
E = 31.68 ft./sec.

Observe what happens if E remains constant at this new value for 1 sec. (see Figure 7).

Increase in Distance = 1 sec. x 31.68 ft./sec.
= 31.68 ft.
Total Distance = 88 ft. + 52.8 ft. + 31.68 ft.
= 172.48

Figure 7 shows that the corrected velocity signal E gets smaller. This means that the total distance traveled will approach some limit as the corrected velocity approaches zero. This limit on total distance should be 220 ft. in accord with the system goal. How can this be determined? One way would be to make calculations by hand like those above for many 1 sec. intervals and check to see if D approaches 200 ft. Since these calculations are very tedious, it would be helpful to write a computer program in BASIC and let the computer become a model of the dynamic feedback system. By studying this model, it can be determined if the system is seeking the goal.

Study the flow chart (Figure 8) to make sure you understand the repetetive nature of the calculations which the computer will make. If you do not understand the flow chart, go back and study the hand calculations shown above.

A computer program and subsequent run for the system are shown in Figure 9. Notice how the system "seeks" the goal. Velocity is corrected until it approaches zero. Distance traveled increases and approaches 220 ft. This is feedback in a dynamic system. Output is measured and used to alter input which produces new output. The system displays goal seeking behavior.

This model could now be compared to the real world and improved so that it more accurately represented the physical system. For example, we could use a time interval T2 of 0.00001 sec.. This would have the effect of smoothing out the curves in Figures 4-7. This changed model would more accurately represent the real world, as automobiles cannot instantaneously change from a velocity of 88 ft./sec. to 52.8 ft./sec. Other changes in the model could be introduced to improve the representation of the real world. In any case, we can now see the importance of feedback in controlling systems.

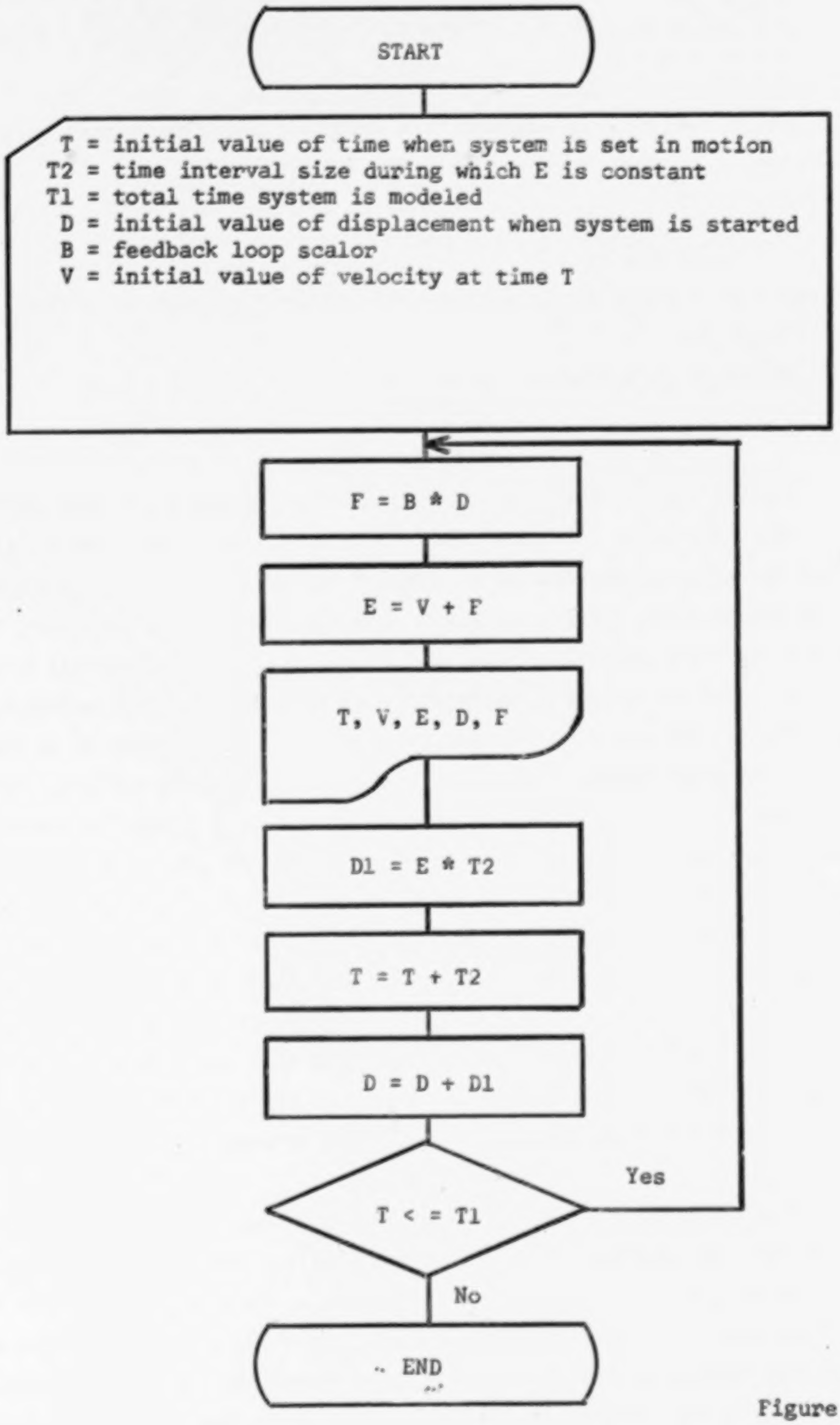

START
T = initial value of time when system is set in motion
T2 = time interval size during which E is constant
T1 = total time system is modeled
D = initial value of displacement when system is started
B = feedback loop scalor
V = initial value of velocity at time T
F = B * D
E = V + F
T, V, E, D, F
D1 = E * T2
T = T + T2
D = D + D1
T < = T1
Yes
No
END

Figure 8

```
FDBMDL     15:27     PX MØN 07/17/67

100    REM D = INITIAL VALUE ØF DISPLACEMENT WHEN SYSTEM IS STARTED
110    REM B = FEEDBACK LØØP SCALØR ØR MULTIPLIER VALUE
120    REM V = INITIAL VALUE ØF VELØCITY AT TIME T
130    REM E = CØRRECTED VELØCITY SIGNAL
140    REM F = FEEDBACK SIGNAL
150    REM D1 = CHANGE IN DISPLACEMENT
160    REM T = INITIAL VALUE ØF TIME WHEN SYSTEM IS SET IN MØTIØN
170    REM T1 = TØTAL TIME SYSTEM IS MØDELED
180    REM T2 = TIME INTERVAL SIZE DURING WHICH E IS CØNSTANT
190 READ T,T2,D,B,V,T1
200 DATA 0,1,0,-0.4,88,20
210 PRINT "T","V","E","D","F"
220 PRINT "SEC","FT/SEC","FT/SEC","FT","FT/SEC"
230 LET F=B*D
240 LET E=V+F
250 PRINT T,V,E,D,F,
260 LET D1-E*T2
270 LET T=T+T2
280 LET D=D+D1
290 IF T<=T1 THEN 230
300 END

RUN
```

```
FDBMDL     15:28     PX MØN 07/17/67

T              V               E                   D              F
SEC            FT/SEC          FT/SEC              FT             FT/SEC
 0              88              88                 0               0
 1              88              52.8               88             -35.2
 2              88              31.68              140.8          -56.32
 3              88              19.008             172.48         -68.992
 4              88              11.4048            191.488        -76.5952
 5              88              6.84288            202.893        -81.1571
 6              88              4.10573            209.736        -83.8943
 7              88              2.46344            213.841        -85.5366
 8              88              1.47806            216.305        -86.5219
 9              88              .886837            217.783        -87.1132
 10             88              .532102            218.67         -87.4679
 11             88              .319261            219.202        -87.6807
 12             88              .191557            219.521        -87.8084
 13             88              .114934            219.713        -87.8851
 14             88              6.89605 E-2        219.828        -87.931
 15             88              4.13764 E-2        219.897        -87.9586
 16             88              2.48258 E-2        219.938        -87.9752
 17             88              1.48954 E-2        219.963        -87.9851
 18             88              8.93736 E-3        219.978        -87.9911
 19             88              5.36239 E-3        219.987        -87.9946
 20             88              3.21746 E-3        219.992        -87.9968

TIME:     1 SECS.
```

Figure 9

A SHORT DESCRIPTION OF A NARRATED SERIES OF SLIDES WHICH PRESENT TWO INTRODUCTORY COMPUTER LESSONS FOR JUNIOR HIGH SCHOOL

Narration and Slides prepared by:

Donald Marcotte
K. Wayne Mahan
Dan Colvin

Photographic assistance and art work provided by:

Jefferson County Instructional Aids Department

The Jefferson County Schools have two small instructional computers (Bitran Six) which are used in eighteen junior high schools to illustrate what a computer is and how it functions. The computers are moved from school to school and also provide motivation for study of numeration systems.

As their project for the NSF Institute Mr. Marcotte, Mr. Mahan, and Mr. Colvin prepared two complete lessons with tape recorded narration and color slides to present to junior high classes.

The first lesson provides a brief introduction to computers, a specific introduction to a "teaching computer" (The Bitran Six), a presentation of the numeration system used by computers, and simple programming procedures for the Bitran Six.

The second lesson presents programs for addition, subtraction, multiplication, and division, and suggests other programs for student work.

Copies of tapes and slides or printed copies of the narration with accompanying sketches of the slides may be obtained at cost from the Jefferson County Public Schools Instructional Aids Center, 7655 West 10th Avenue, Lakewood, Colo. 80215.

A COMPUTER ORIENTED APPROACH TO LINEAR EQUATIONS IN ONE UNKNOWN

by Robert A. Norton and Roe E. Willis

Today's junior high school student is destined to have some contact with computers and computer programming before high school graduation. This is the impetus for writing this text.

The basic tool in this text is the flow chart which is used as a means of teaching solutions of single variable linear equations. The text is written for the student to read, reason through, and work out the exercises. Eighth or ninth grade students are best suited for the content, although an aggresive seventh grader could be expected to succeed.

The flow chart is the basic step for computer programming. The self-teaching properties of the flow chart coupled with its motivational value will give the student a good basic foundation.

This is not intended as a package text. It is a preliminary step in using a computer oreinted concept to teach a section of elementary algebra. Other topics readily lend themselves to this technique. Inequalities, systems of linear equations, and quadratic equations are such examples.

The Flow Chart

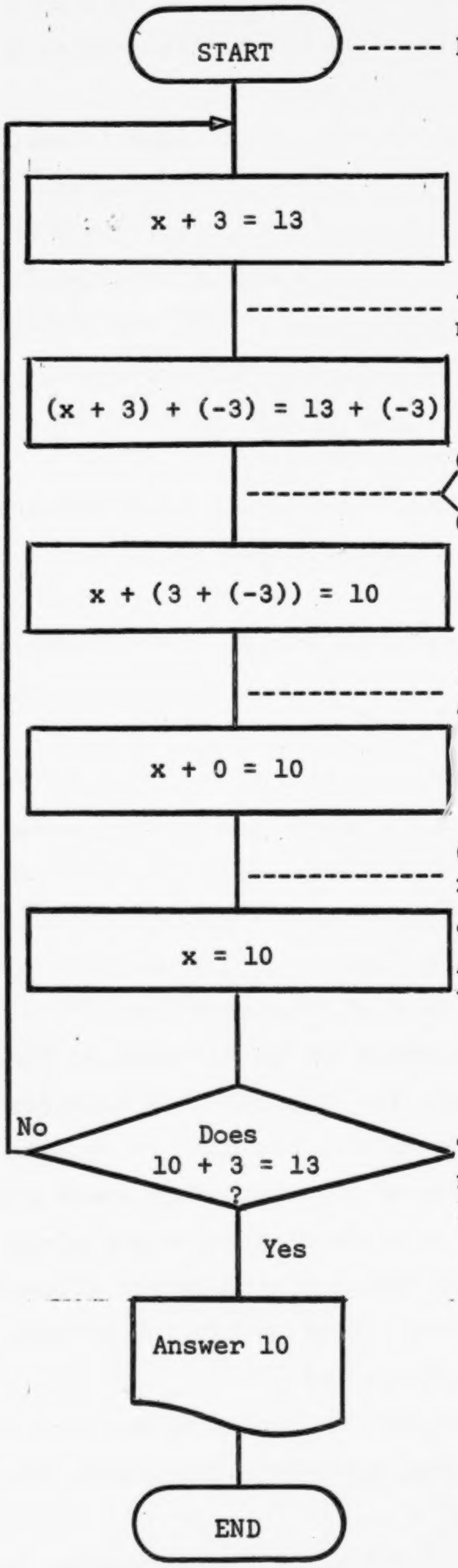
The equation $x + 3 = 13$ is asking you to find a number to replace x which will make the sentence (equation) true. The obvious answer is 10. The aim here is learning to solve much more difficult problems. This requires an organized approach to solving a simple problem and will lead to more complicated problems using this basic approach.

Figure 1 is the flow chart solution of this equation.

There are five mathematical sentences in the rectangular boxes. If x is replaced by 10 in each of them, the equations are satisfied or true. When two or more sentences have exactly the same truth set, the sentences are equivalent. We have a way of making certain every step we make gives an equivalent equation. The properties which are given in Appendix 1 are the necessary tools for this. A step is not allowed unless there is a property to support it. When we have to justify or give a reason for each step this is mathematics.

From here on you are required to justify each step you make by stating the property used. Remember equivalent equations mean they have the same truth set.

Figure 2 shows the first solution and gives the properties for each step.



Read the problem and plan your solution.

-3 is added to both sides. Any other number would have wasted our time.

ON THE LEFT SIDE 3 and -3 are grouped for addition.
ON THE RIGHT SIDE 13 and -3 are added.

3 and -3 added give 0. This is why we added -3.

0 added to x gives x regardless of x's value.

This is the simplest form of any equation. All steps are planned from the beginning toward this final equation.

This is the check. If the value does not check go through the solution again. If it does check this is the answer.

Figure 1

A lot of work for a simple problem? Yes, it is; but now a pattern is set. Every problem like this is solved by the same steps.

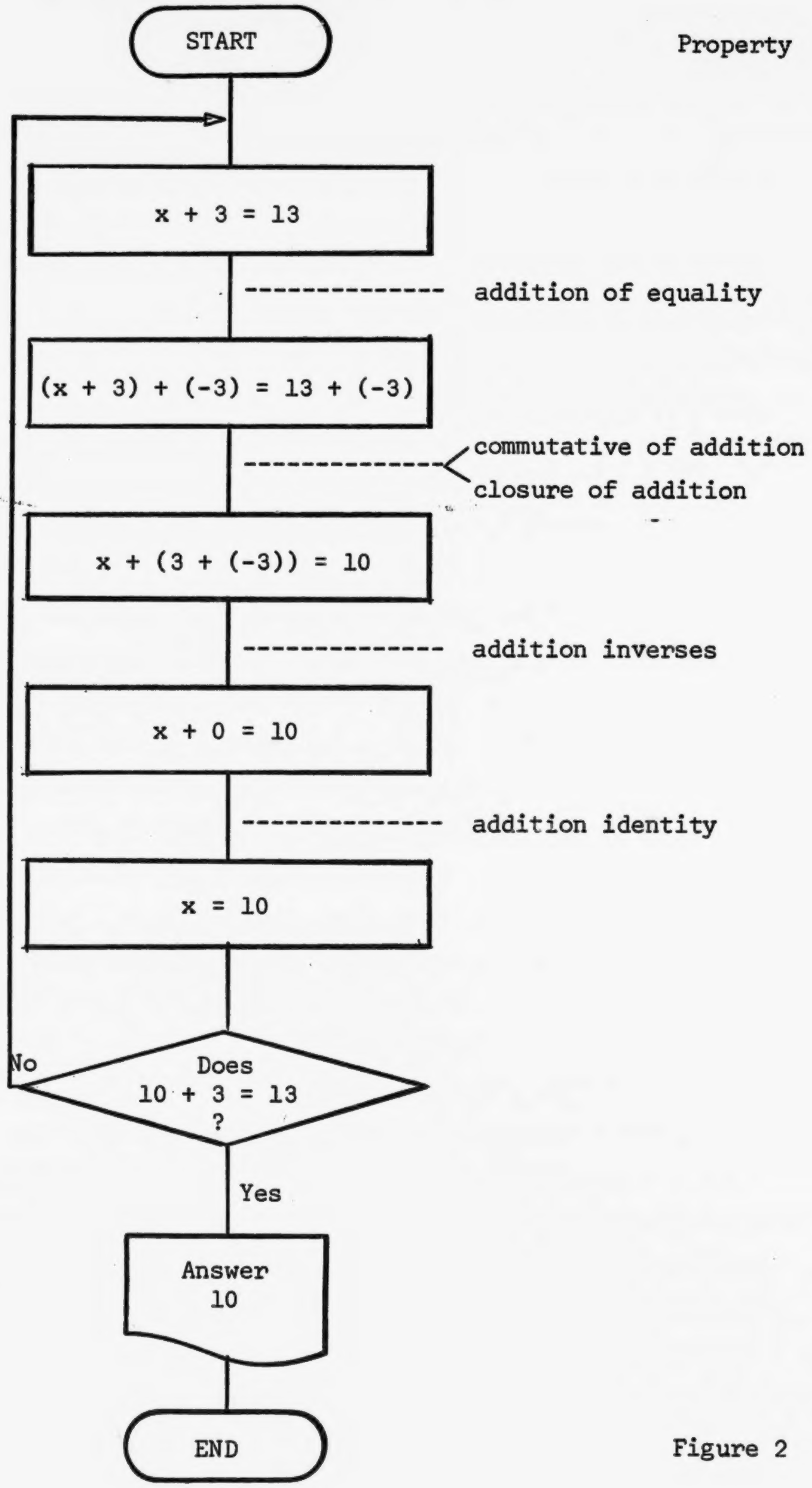

START
Property
x + 3 = 13
addition of equality
(x + 3) + (-3) = 13 + (-3)
commutative of addition
closure of addition
x + (3 + (-3)) = 10
addition inverses
x + 0 = 10
addition identity
x = 10
No
Does
10 + 3 = 13
?
Yes
Answer
10
END

Figure 2

The foregoing problem involved only properties of addition. Here is a similar problem involving only multiplication. Solve -3x = 24, and give the properties for each step. Refer to Figure 3.

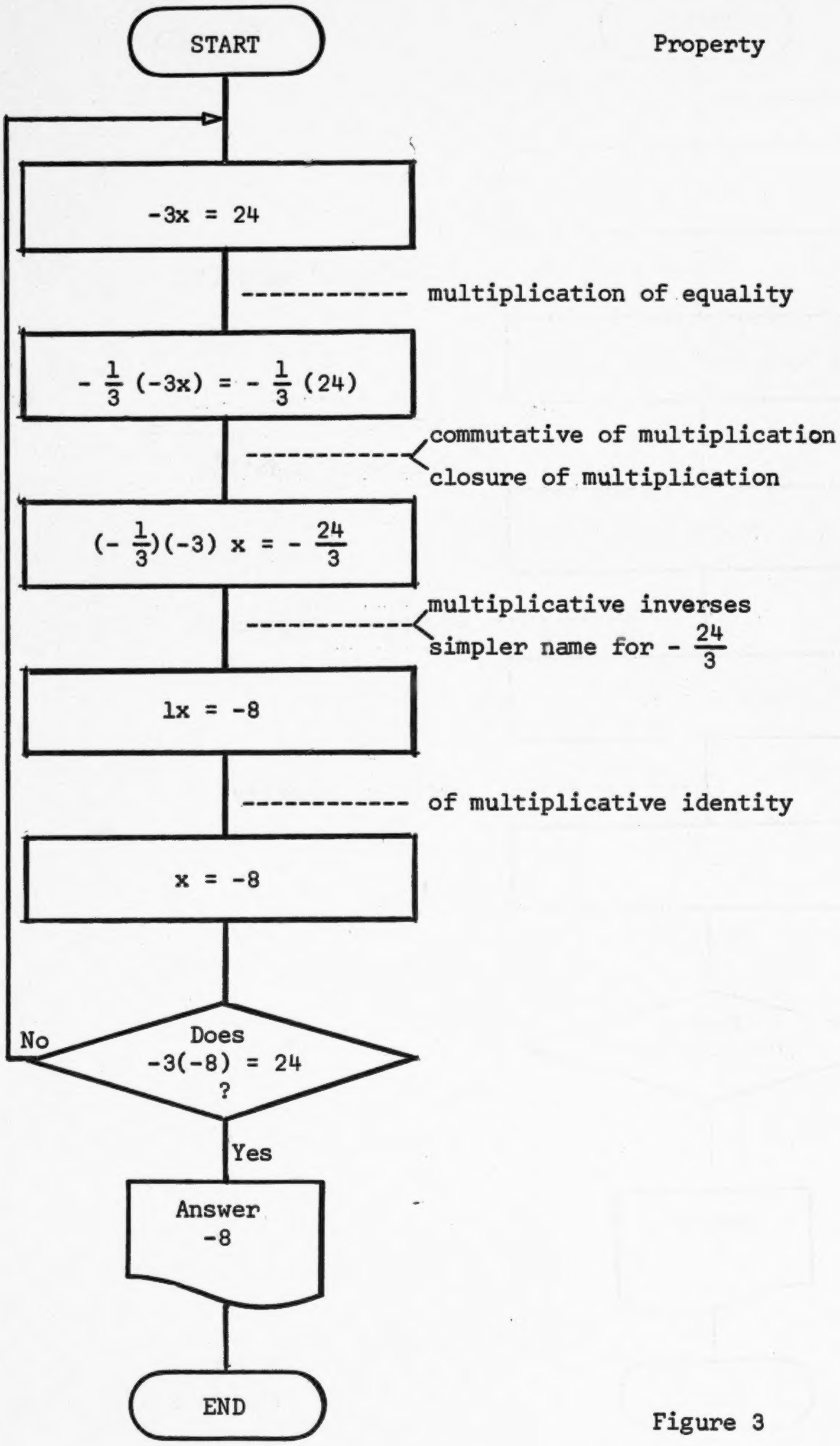

START
Property
$-3x = 24$
multiplication of equality
$-\frac{1}{3}(-3x) = -\frac{1}{3}(24)$
commutative of multiplication
closure of multiplication
$(-\frac{1}{3})(-3)x = -\frac{24}{3}$
multiplicative inverses
simpler name for $-\frac{24}{3}$
$1x = -8$
of multiplicative identity
$x = -8$
No
Does $-3(-8) = 24$?
Yes
Answer -8
END

Figure 3

Exercises I

Flow chart the solutions, and give the properties for each step in the solutions of the following problems.

a) $3 = x - 18$ b) $4x = -22$ c) $c - 4x = 9$

If your flow chart solution does not agree with the answer given, carefully check and compare your steps. You are developing two techniques: 1) steps in solving equations and 2) an understanding of why we can make each step.

We slipped in a property in Problem Ia which had not been mentioned. The properties only provide for addition and multiplication. Subtraction is defined with addition. For example, $3 - 8$ is the same as $3 + (-8)$. Division is defined in a similar way -- $9 \div (-3)$ is the same as $9(-\frac{1}{3})$. This provides us with a well defined set of properties. These are sufficient for solving any linear equation using rational numbers.

The next equation to be considered is one like Ic involving both addition and multiplication properties. The solution of $13 = 4 - 5x$ is shown in Figure 4.

In checking the answer, the order of operations is: 1) Do all operations within the parentheses first, 2) all multiplications, and 3) all additions.

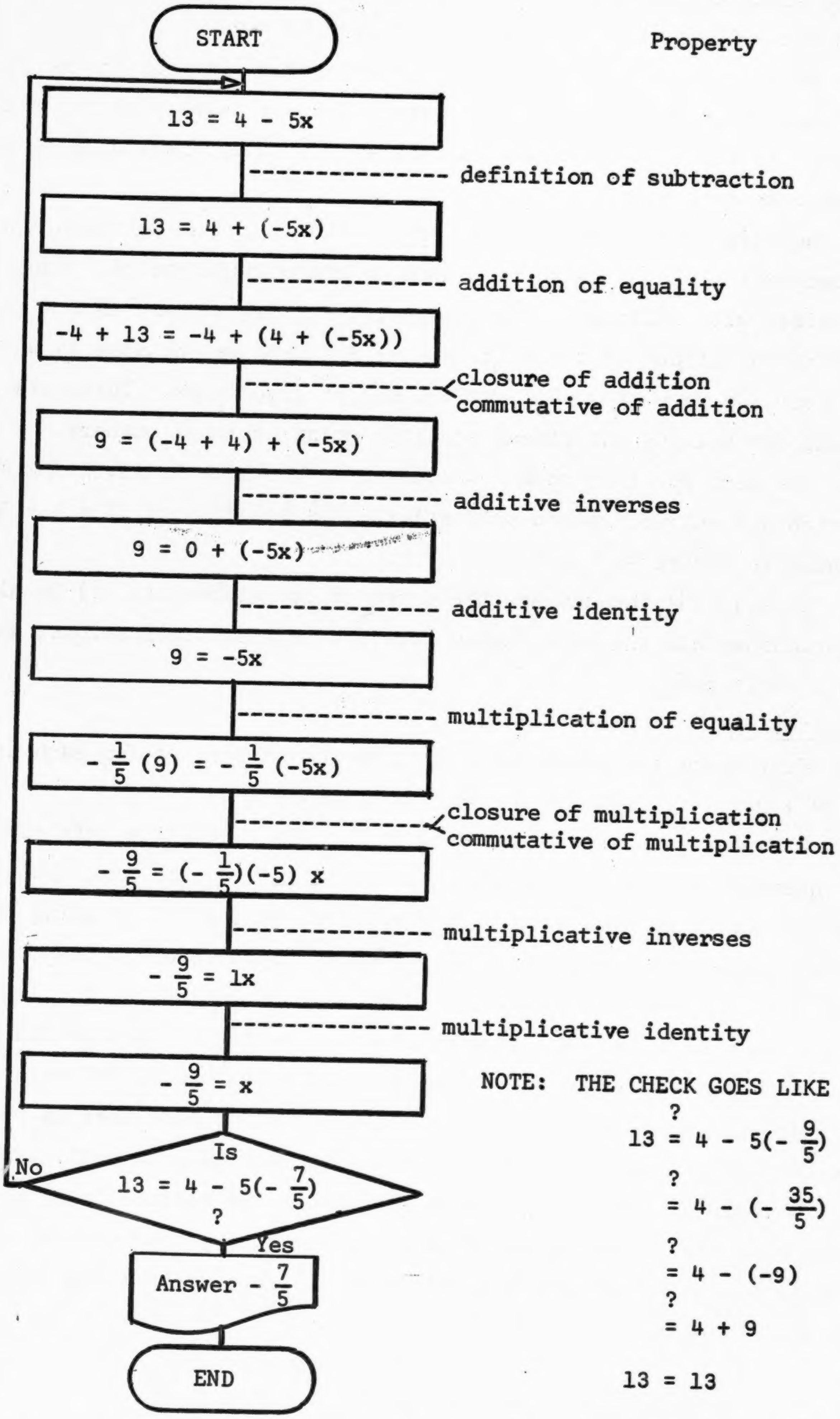
Exercises II

Flow chart the solutions, and give the properties for each step.

a) $3x + 2 = 7$ b) $9 = 5 - \frac{x}{2}$ c) $2\frac{1}{2}x - 17 = 3$

The next problem to be solved is $2x + 3x = 20$. The left side of the equation $2x + 3x$ can be rewritten as $(2 + 3)x$. The distributive property justifies this step. It simplifies further to $5x$ using the closure property of addition. This problem could occur in the following forms and have the same basic solution: 1) $2x + 3x = 20$, 2) $20 - 3x = 2x$, and 3) $3x = 20 - 2x$. The flow chart solution for 2) is shown in Figure 5.

Now that the distributive property is explained, START means to look for all terms containing the variable. x has been used as our variable throughout, but y, z, etc., could have been used as well. In planning the solution ahead, all terms containing the variable must end up on the same side. Use of the distributive property then enables these terms to be collected into a single term. Figure 6 shows the solution of $3x - 13 = 7x + 4x$.


START
Property
13 = 4 - 5x
definition of subtraction
13 = 4 + (-5x)
addition of equality
-4 + 13 = -4 + (4 + (-5x))
closure of addition
commutative of addition
9 = (-4 + 4) + (-5x)
additive inverses
9 = 0 + (-5x)
additive identity
9 = -5x
multiplication of equality
$-\frac{1}{5}(9) = -\frac{1}{5}(-5x)$
closure of multiplication
commutative of multiplication
$-\frac{9}{5} = (-\frac{1}{5})(-5)x$
multiplicative inverses
$-\frac{9}{5} = 1x$
multiplicative identity
$-\frac{9}{5} = x$
Is
$13 = 4 - 5(-\frac{7}{5})$
?
No
Yes
Answer $-\frac{7}{5}$
END

NOTE: THE CHECK GOES LIKE THIS

$$13 \overset{?}{=} 4 - 5(-\frac{9}{5})$$

$$\overset{?}{=} 4 - (-\frac{35}{5})$$

$$\overset{?}{=} 4 - (-9)$$

$$\overset{?}{=} 4 + 9$$

$$13 = 13$$

Figure 4

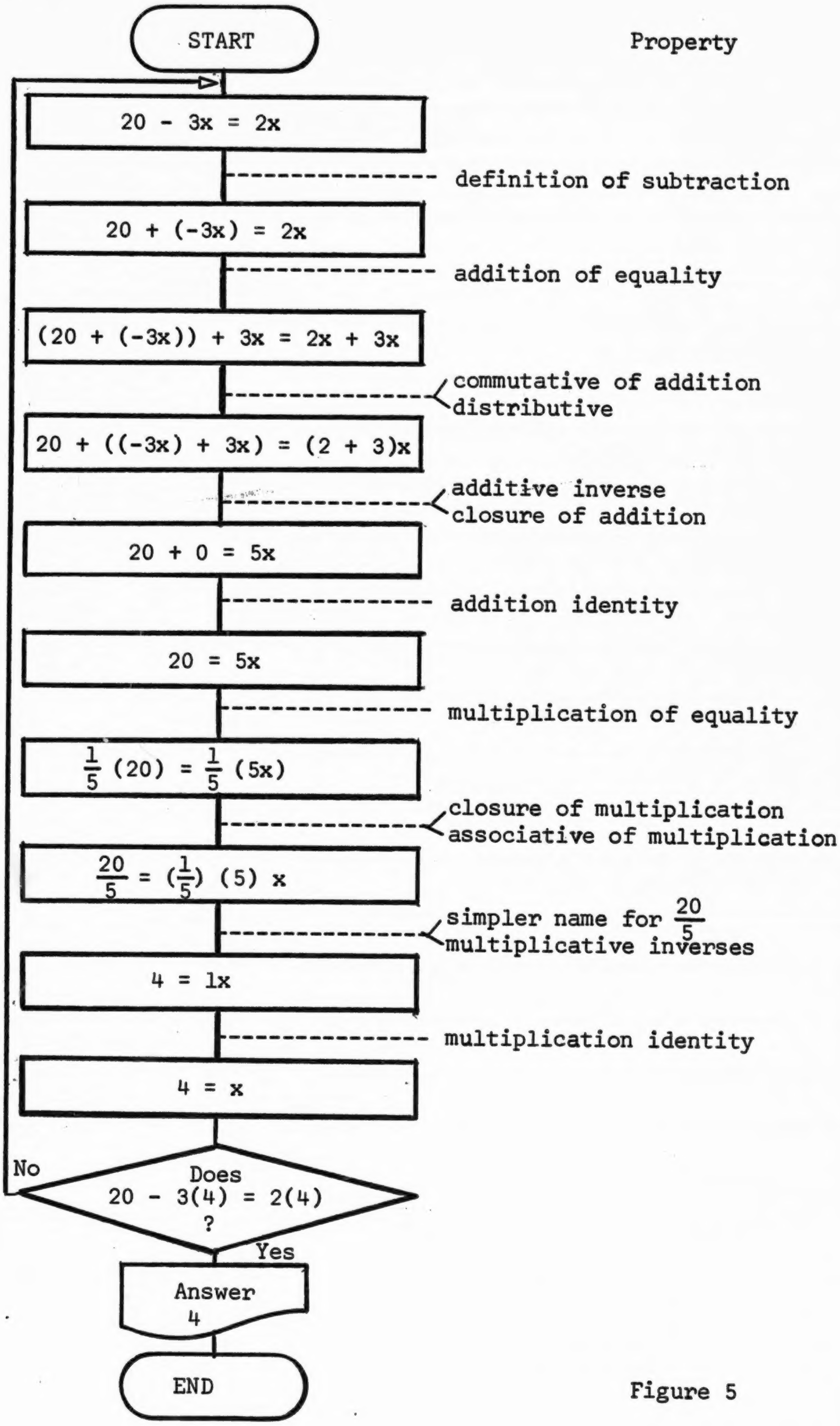
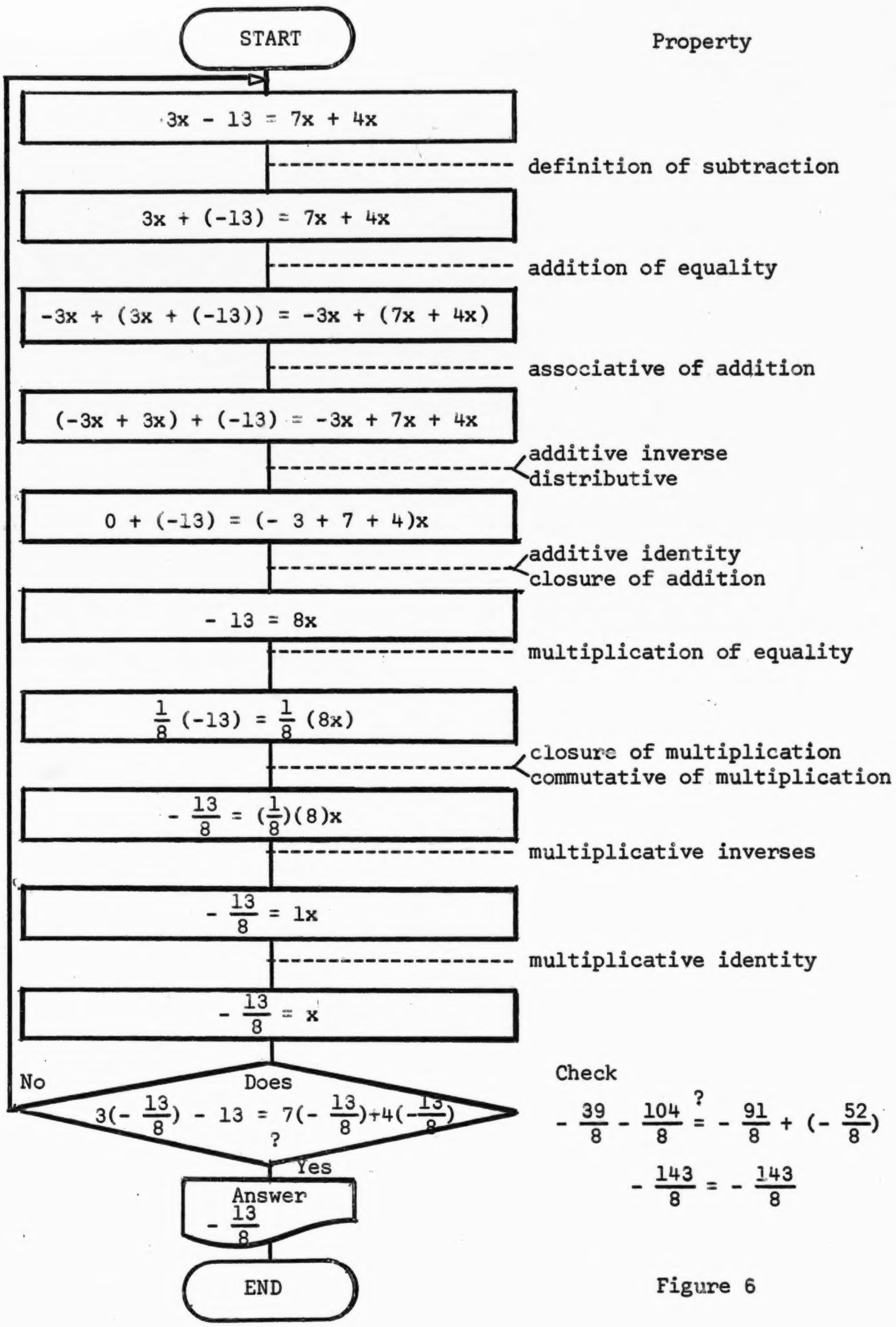

START
Property
20 - 3x = 2x
definition of subtraction
20 + (-3x) = 2x
addition of equality
(20 + (-3x)) + 3x = 2x + 3x
commutative of addition
distributive
20 + ((-3x) + 3x) = (2 + 3)x
additive inverse
closure of addition
20 + 0 = 5x
addition identity
20 = 5x
multiplication of equality
$\frac{1}{5}(20) = \frac{1}{5}(5x)$
closure of multiplication
associative of multiplication
$\frac{20}{5} = (\frac{1}{5})(5)x$
simpler name for $\frac{20}{5}$
multiplicative inverses
4 = 1x
multiplication identity
4 = x
No
Does 20 - 3(4) = 2(4) ?
Yes
Answer 4
END

Figure 5


START
Property
3x - 13 = 7x + 4x
definition of subtraction
3x + (-13) = 7x + 4x
addition of equality
-3x + (3x + (-13)) = -3x + (7x + 4x)
associative of addition
(-3x + 3x) + (-13) = -3x + 7x + 4x
additive inverse
distributive
0 + (-13) = (- 3 + 7 + 4)x
additive identity
closure of addition
- 13 = 8x
multiplication of equality
1/8 (-13) = 1/8 (8x)
closure of multiplication
commutative of multiplication
- 13/8 = (1/8)(8)x
multiplicative inverses
- 13/8 = 1x
multiplicative identity
- 13/8 = x
No
Does
3(- 13/8) - 13 = 7(- 13/8)+4(- 13/8)
?
Yes
Answer
- 13/8
END

Check

$$-\frac{39}{8} - \frac{104}{8} \overset{?}{=} -\frac{91}{8} + \left(-\frac{52}{8}\right)$$

$$-\frac{143}{8} = -\frac{143}{8}$$

Figure 6

An equation like $3(x + 2) = 21$ involves the distributive property in reverse order. Refer to Figure 7.

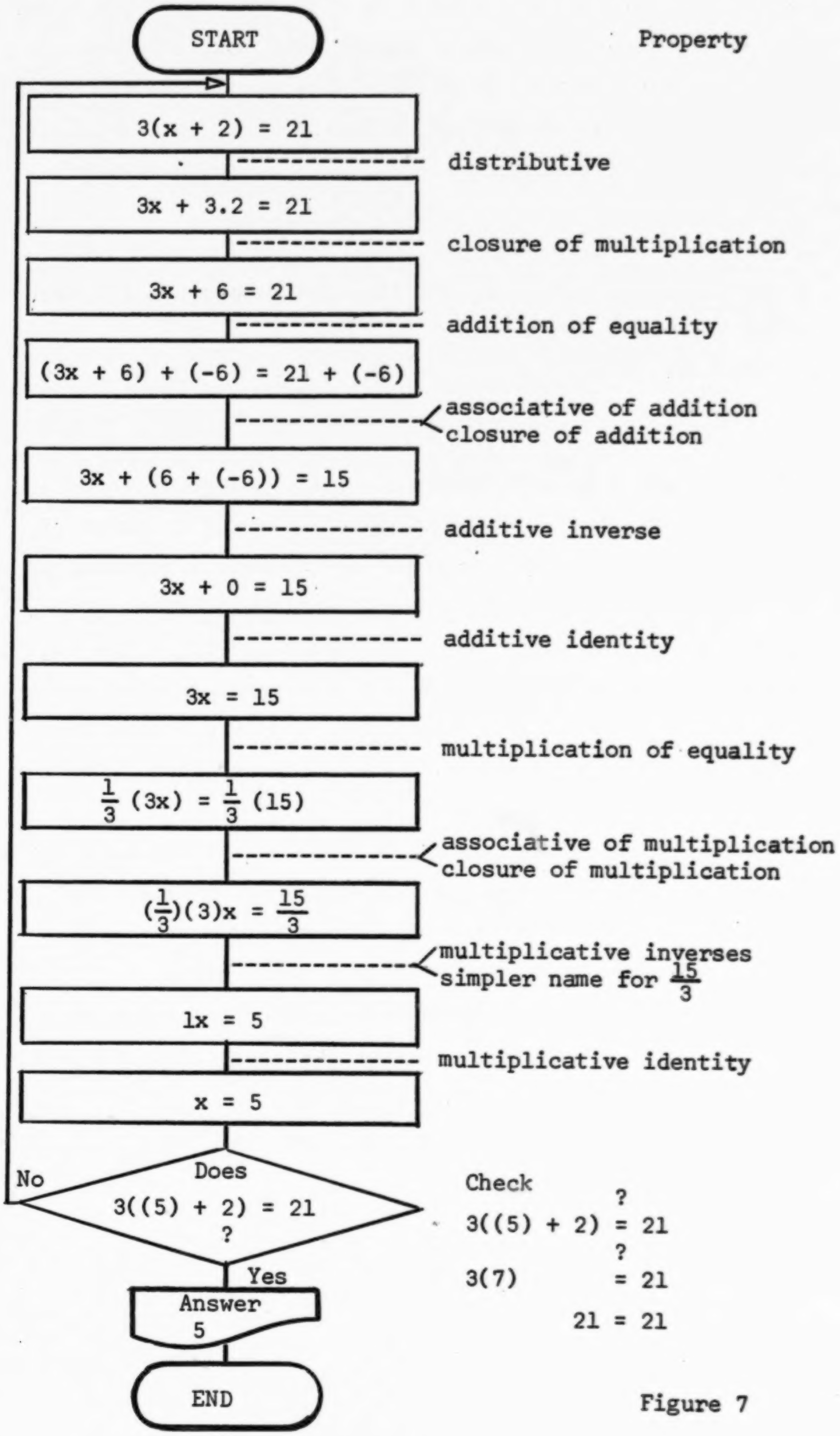

START
Property
3(x + 2) = 21
distributive
3x + 3.2 = 21
closure of multiplication
3x + 6 = 21
addition of equality
(3x + 6) + (-6) = 21 + (-6)
associative of addition
closure of addition
3x + (6 + (-6)) = 15
additive inverse
3x + 0 = 15
additive identity
3x = 15
multiplication of equality
$\frac{1}{3}(3x) = \frac{1}{3}(15)$
associative of multiplication
closure of multiplication
$(\frac{1}{3})(3)x = \frac{15}{3}$
multiplicative inverses
simpler name for $\frac{15}{3}$
1x = 5
multiplicative identity
x = 5
No
Does 3((5) + 2) = 21 ?
Yes
Answer 5
END
Check
$3((5) + 2) \overset{?}{=} 21$
$3(7) \overset{?}{=} 21$
$21 = 21$

Figure 7

It should be mentioned at this point, for $3 + 7x + 8 = 21$, 3 and 8 should be associated and added into a single term. The result would be $7x + 11 = 21$ or $11 + 7x = 21$. 3, 8, 21 are numerical terms and 7x a variable term. Always plan your solution ahead with two objectives in mind: 1) Collect all numerical terms into a single term, and collect all variable terms into a single term; and 2) pick the shortest route with the least number of steps.

EXERCISES III

Flow chart the solutions, and give the properties for each step in the following exercises:

a) $3(x - 4) = 21$ b) $4x - 8x = -72$ c) $4 - 3x = 2x + 9$

d) $\frac{x}{3} - 2 = 4(2 - \frac{x}{6})$ e) $2(x - 3) = 7 - 2x$

These few pages have exposed you to a computer oriented concept -- the flow chart. The flow chart is a way or means of organizing a solution. More important is the actual mathematics involved. The solution appears to be our main objective, but like any problem if the steps have no support or reason the solution means nothing. Problem solution is formally named applied mathematics. One branch can be emphasized more than the other, but it is impossible to study one without in some way considering the other.

Merging of applied and pure mathematics has given us today's high speed computers. The applied mathematician provides the steps it must go through, while the pure mathematician provides the logic or reasoning required to achieve these steps.

Once you have successfully understood the properties and their use, it is to your benefit to drop this formality and develop efficient equation solution methods for yourself.

Consider Thirds:

$$\frac{x}{3} - 2 = 4(2 - \frac{x}{6})$$

$$\frac{x}{3} - 2 = 8 - \frac{2}{3}x$$

$$(\frac{2}{3}x + 2) + \frac{x}{3} - 2 = (\frac{2}{3}x + 2) + 8 - \frac{2}{3}x$$

$$1x = 10$$

$$x = 10$$

Check

$$\frac{10}{3} - 2 \overset{?}{=} 4(2 - \frac{10}{6})$$

$$\frac{10}{3} - \frac{6}{3} = 4(\frac{12}{6} - \frac{10}{6})$$

$$\frac{4}{3} = 4(\frac{2}{6})$$

$$\frac{4}{3} = \frac{4}{3}$$

This is a quick and efficient method. Why all the careful step-by-step reasoning? If every problem you encounter is no more difficult or different than these, there would be no purpose. However, a new and completely different problem can be successfully solved if a step-by-step solution is organized with a property for each step.

The computer has a language it understands. If the engineer can break his problem down to the computer's language, the solution is performed in seconds compared to days. The flow chart is the first step in organizing a problem before putting it into computer language.

Already computers have reached the speed of light, which isn't fast enough for complex weather problem solutions to be ready for the evening news. Finding different methods for solving the same problem and continued redesigning of the computer network will keep the two branches of mathematicians busy for some time.

Don't forget -- they had to begin where you are now.

Appendix I

Properties and Definitions Used to Establish Simpler Equivalent Equations in the Rational Number Field

Properties in the Rational Number Field

1) Closure property of addition: If a and b are any rational numbers, then a + b is exactly one rational number.
Examples: $2 + \frac{2}{3} = 2\frac{2}{3}$; $4 + 10 = 14$

2) Closure property of multiplication: If a and b are any rational numbers, then (a)(b) is exactly one rational number.
Examples: $(2)(\frac{1}{3}) = \frac{2}{3}$; $(\frac{2}{3})(\frac{5}{7}) = \frac{10}{21}$; $(3)(2) = 6$

3) Commutative property of addition: If a and b are any rational numbers, then a + b = b + a.
Examples: $\frac{2}{3} + \frac{1}{5} = \frac{1}{5} + \frac{2}{3}$; $5 + 7 = 7 + 5$

4) Commutative property of multiplication: If a and b are any rational numbers, then (a)(b) = (b)(a).
Examples: $(2)(3) = (3)(2)$; $(\frac{5}{3})(4) = (4)(\frac{5}{3})$

5) Associative property of addition: If a, b, and c are any rational numbers, then (a + b) + c = a + (b + c).
Example: $(2 + 5) + 4 = 2 + (5 + 4)$
Verify: $7 + 4 = 2 + 9$
$11 = 11$ True

6) Associative property of multiplication: If a, b, and c are any rational numbers, then ((a)(b))c = a((b)(c)).
Example: $((2)(3))(4) = (2)((3)(4))$
Verify: $(6)(4) = (2)(12)$
$24 = 24$ True

7) Identity property of addition: If a is any rational number, then a + 0 = 0 + a = a.
Examples: $5 + 0 = 5$; $0 + \frac{2}{3} = \frac{2}{3}$

i.e., zero is the identity element for addition.

8) Identity property of multiplication: If a is any rational number, then (a)(1) = (1)(a) = a.
Examples: $(\frac{22}{7})(1) = \frac{22}{7}$; $(1)(2) = 2$

i.e., one is the identity element for multiplication.

9) Inverse property of addition: If a is any rational number and b is its opposite (additive inverse), then $a + b = 0$.
Examples: $2 + (-2) = 0$; $(-\frac{5}{3}) + \frac{5}{3} = 0$

10) Inverse property of multiplication: If a is any rational number except zero and b is its multiplicative inverse, then $(a)(b) = 1$.
Examples: $(5)(\frac{1}{5}) = 1$; $(2\frac{1}{2})(\frac{2}{5}) = 1$

11) Distributive property of multiplication over addition: If a, b, and c are any rational numbers, then $a(b + c) = (a)(b) + (a)(c)$.
Example: $3(4 + 5) = (3)(4) + (3)(5)$
Verify: $3(9) = 12 + 15$
$27 = 27$ True

12) Multiplication property of zero: If a is any rational number, then $(a)(0) = 0$.
Examples: $(0)(2) = 0$; $(\frac{2}{3})(0) = 0$

13) Addition property of equality: If a, b, and c are any rational numbers and $a = b$, then $a + c = b + c$ and is equivalent to $a = b$.
Examples: $2 = \frac{4}{2}$ and $2 + 5 = \frac{4}{2} + 5$ are equivalent.
$x = 5$ and $x + 3 = 5 + 3$ are equivalent.

14) Multiplication property of equality: If a, b, and c are any rational numbers, $c \neq 0$, and $a = b$, then $(a)(c) = (b)(c)$ and is equivalent to $a = b$.
Examples: $5 = \frac{15}{3}$ and $(5)(7) = (\frac{15}{3})(7)$ are equivalent.
$x = 5$ and $3x = 15$ are equivalent.

__Definitions in the Rational Number Field__

1) Solution set: replacement values for the variable which make the statement true.

2) Division by zero: undefined in the field of rational numbers.

3) Equivalent statements or equations: statements which have the same solution set.

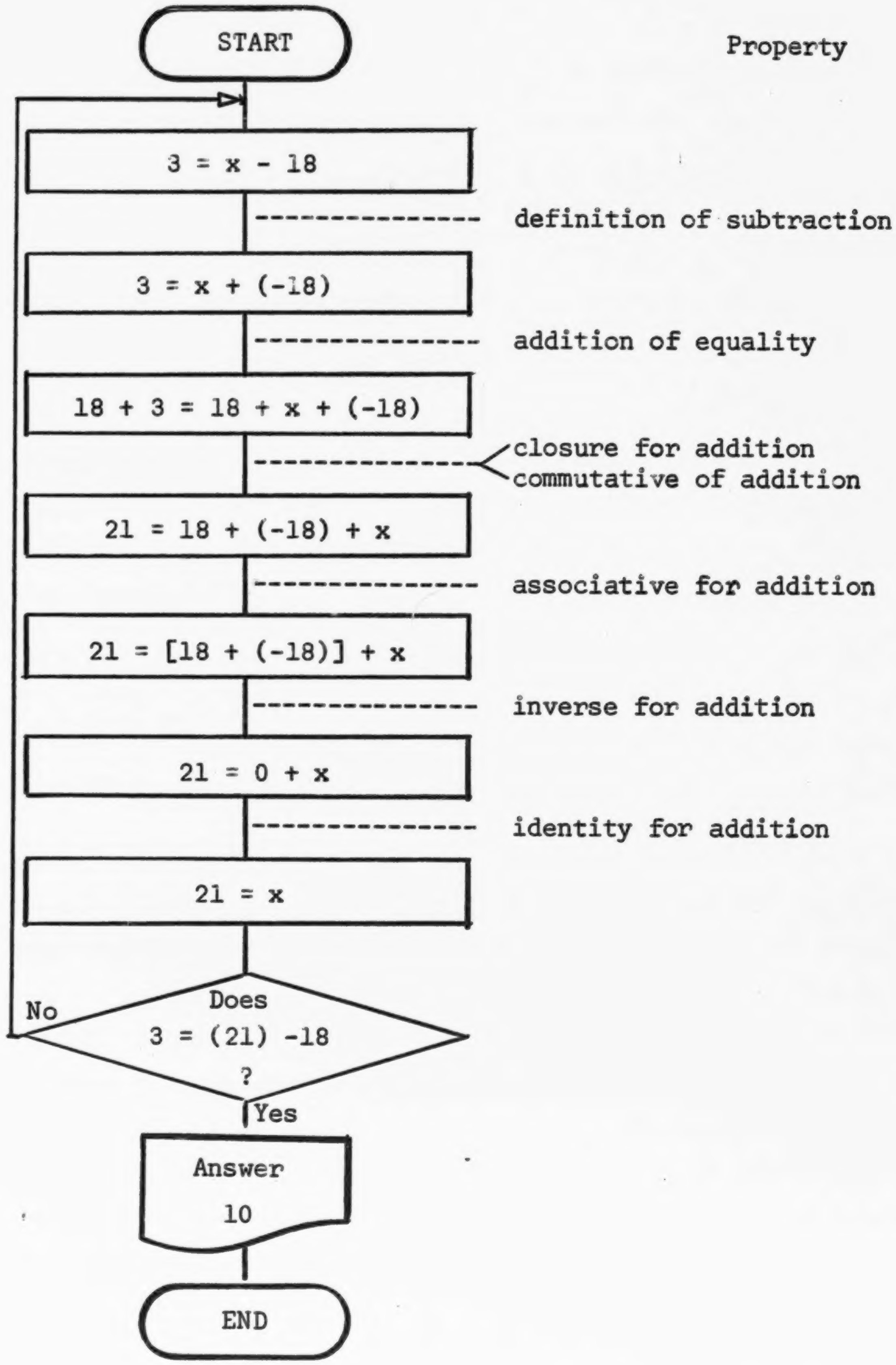
4) Subtraction defined in terms of addition: $a - b = a + (-b)$
Examples: $2 - 4 = 2 + (-4)$; $\frac{3}{8} - \frac{2}{3} = \frac{3}{8} + (-\frac{2}{3})$

5) Division defined in terms of multiplication: $\frac{a}{b} = (\frac{1}{b})(a)$
Examples: $\frac{2}{5} = (\frac{1}{5})(2)$; $\frac{x}{3} = (\frac{1}{3})(x)$

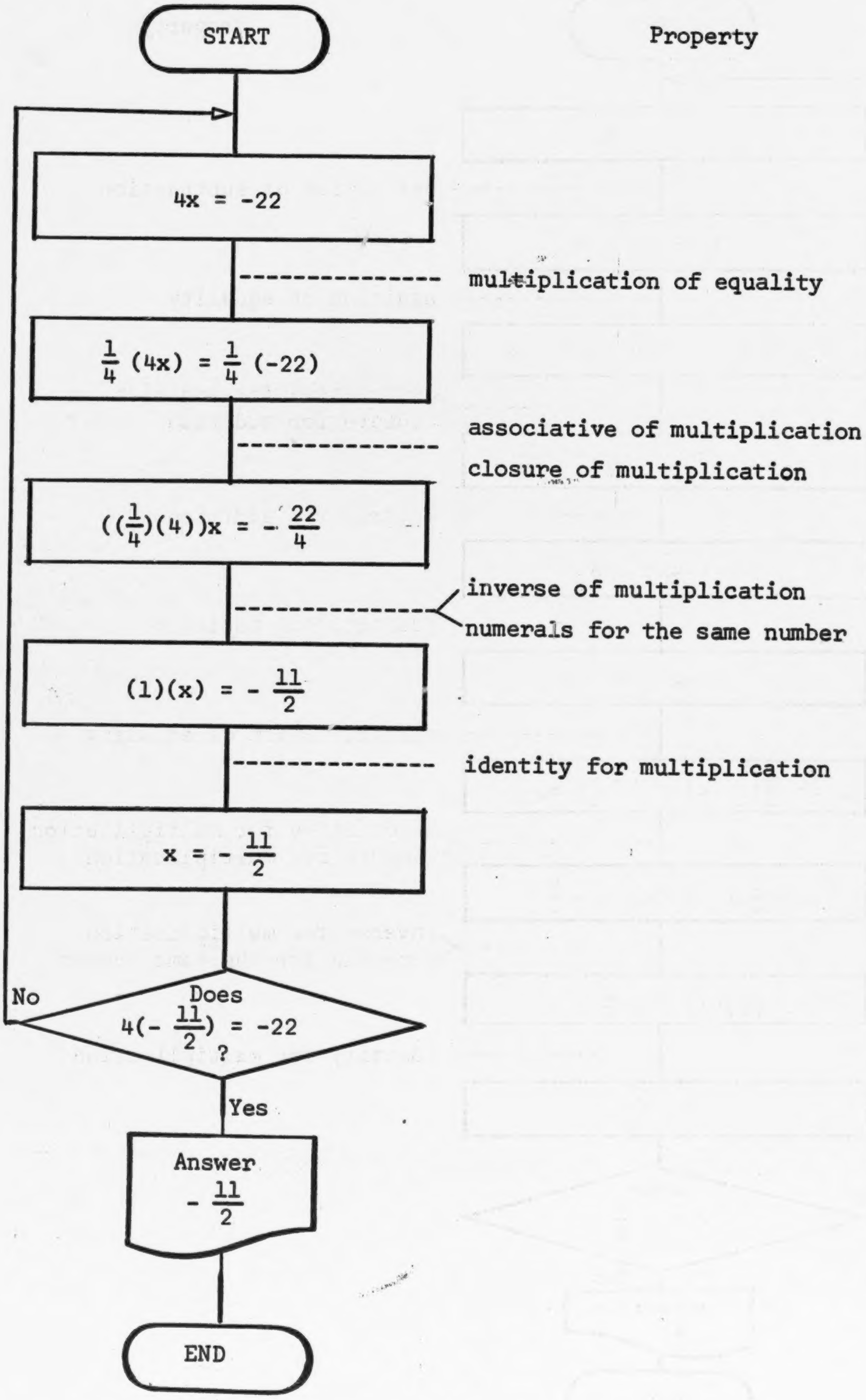
Appendix II

ANSWERS

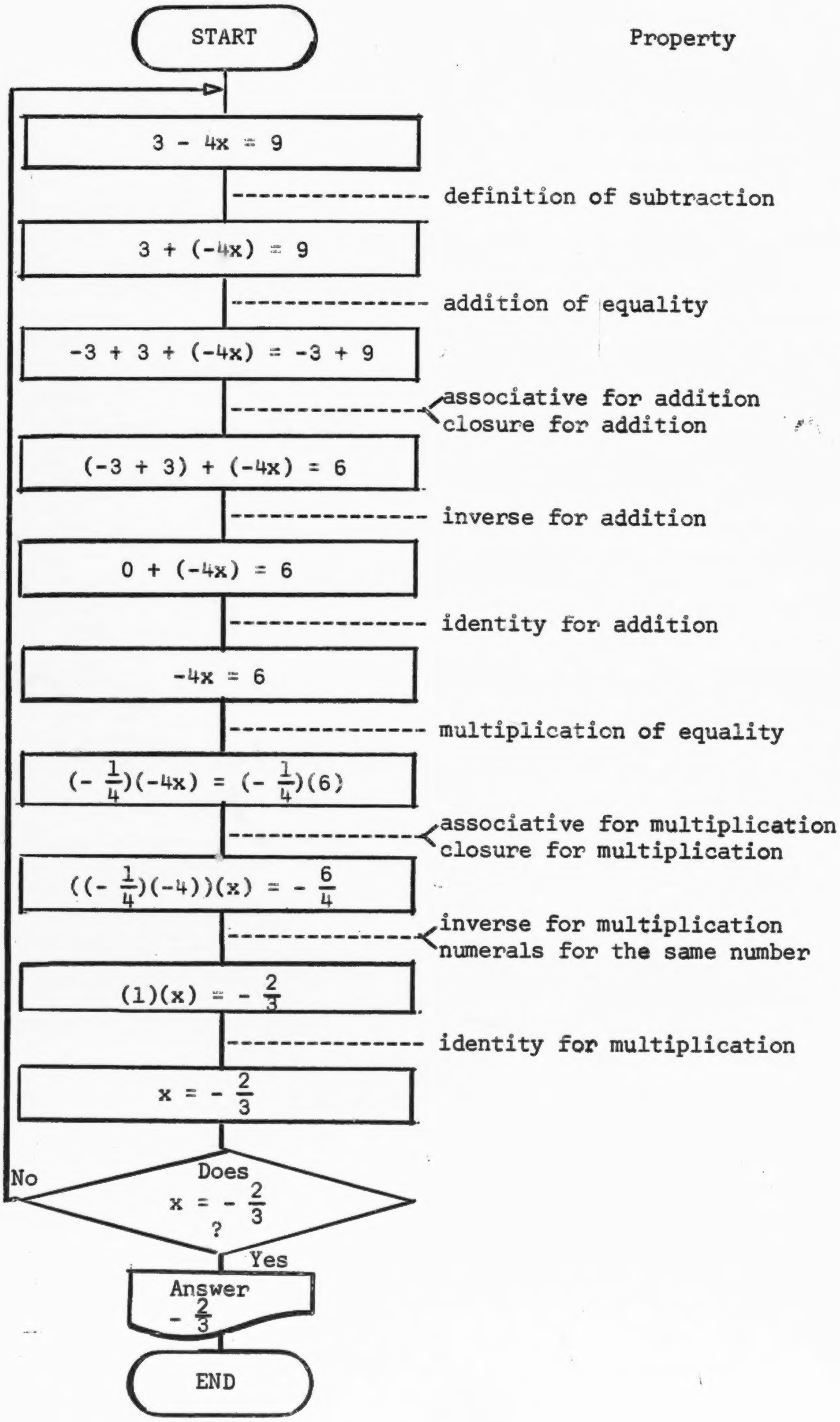
1a)


START
Property
3 = x - 18
definition of subtraction
3 = x + (-18)
addition of equality
18 + 3 = 18 + x + (-18)
closure for addition
commutative of addition
21 = 18 + (-18) + x
associative for addition
21 = [18 + (-18)] + x
inverse for addition
21 = 0 + x
identity for addition
21 = x
No
Does
3 = (21) -18
?
Yes
Answer
10
END

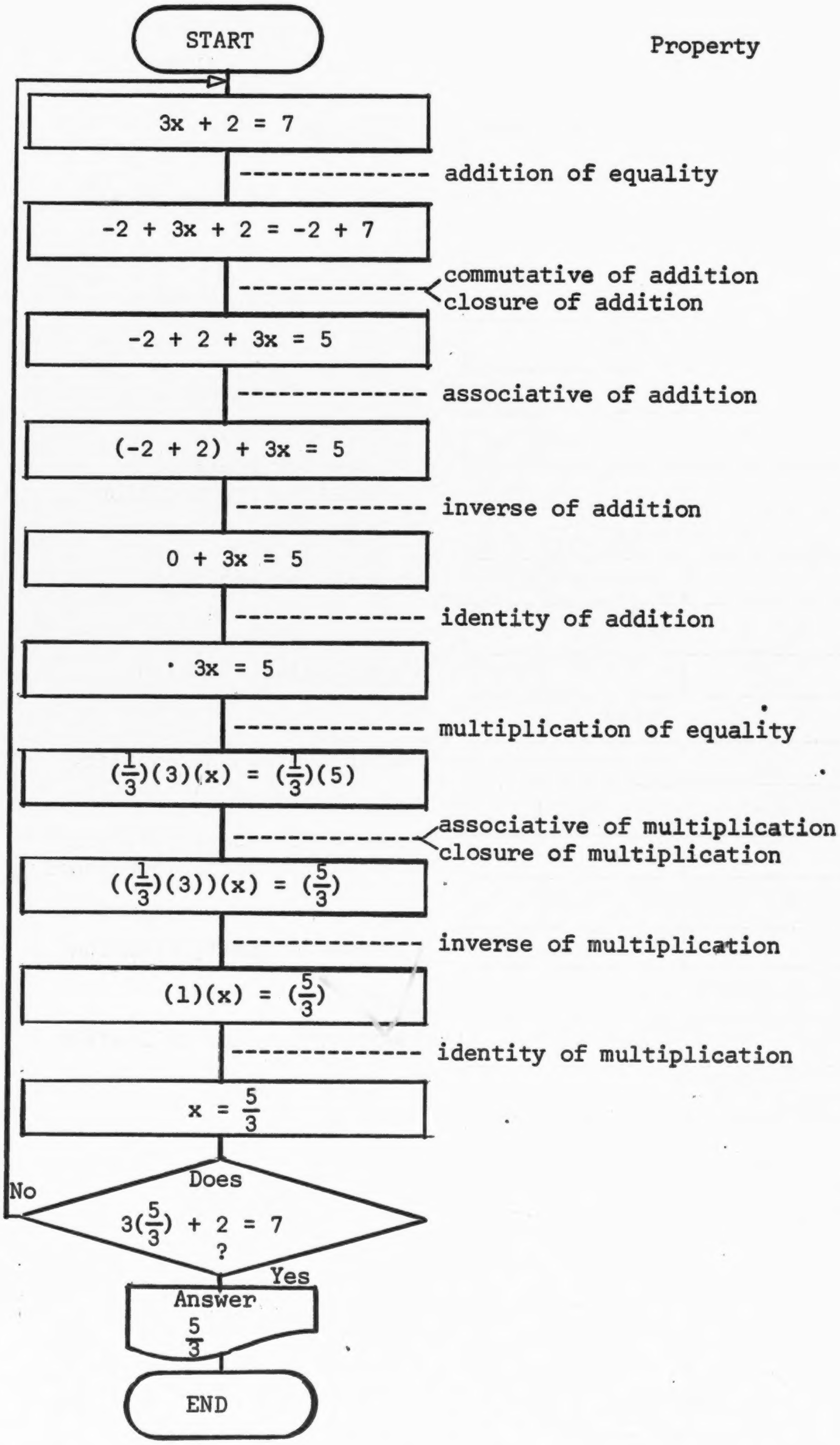
1b)


START
Property
4x = -22
multiplication of equality
$\frac{1}{4}(4x) = \frac{1}{4}(-22)$
associative of multiplication
closure of multiplication
$((\frac{1}{4})(4))x = -\frac{22}{4}$
inverse of multiplication
numerals for the same number
$(1)(x) = -\frac{11}{2}$
identity for multiplication
$x = -\frac{11}{2}$
No
Does $4(-\frac{11}{2}) = -22$?
Yes
Answer $-\frac{11}{2}$
END

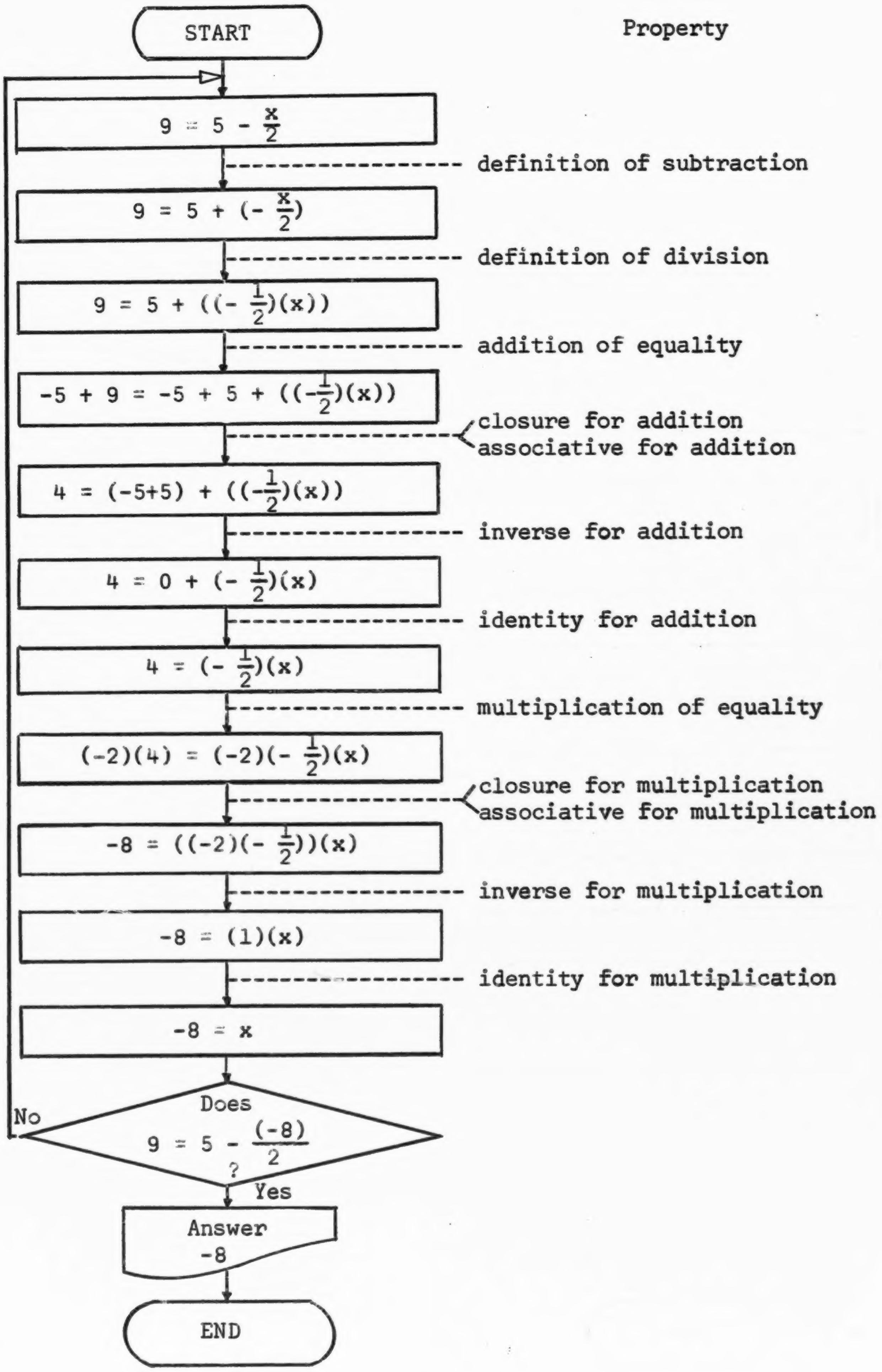
1c)


START
Property
$3 - 4x = 9$
definition of subtraction
$3 + (-4x) = 9$
addition of equality
$-3 + 3 + (-4x) = -3 + 9$
associative for addition
closure for addition
$(-3 + 3) + (-4x) = 6$
inverse for addition
$0 + (-4x) = 6$
identity for addition
$-4x = 6$
multiplication of equality
$(-\frac{1}{4})(-4x) = (-\frac{1}{4})(6)$
associative for multiplication
closure for multiplication
$((-\frac{1}{4})(-4))(x) = -\frac{6}{4}$
inverse for multiplication
numerals for the same number
$(1)(x) = -\frac{2}{3}$
identity for multiplication
$x = -\frac{2}{3}$
No
Does $x = -\frac{2}{3}$?
Yes
Answer $-\frac{2}{3}$
END

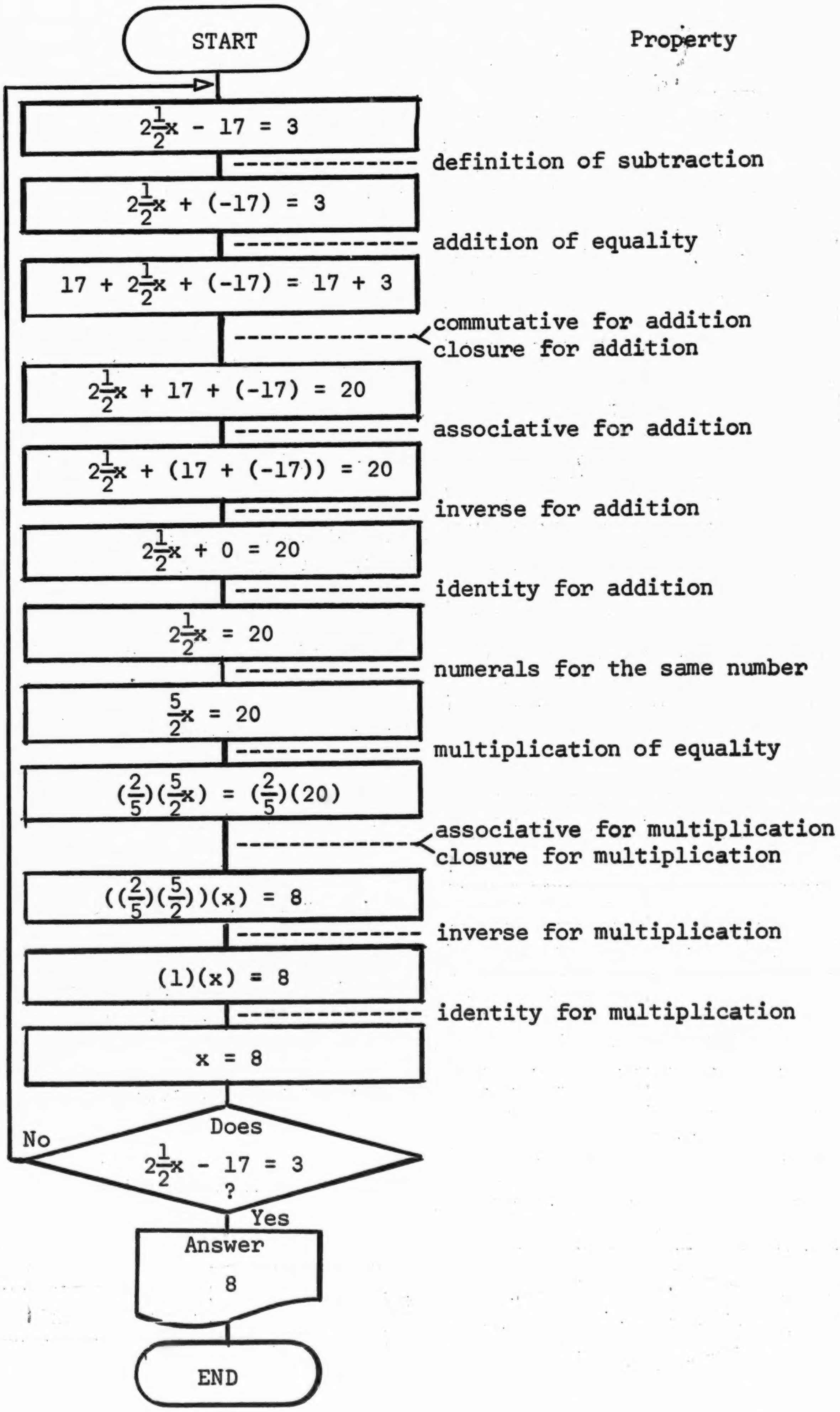
2a)


START
Property
$3x + 2 = 7$
addition of equality
$-2 + 3x + 2 = -2 + 7$
commutative of addition
closure of addition
$-2 + 2 + 3x = 5$
associative of addition
$(-2 + 2) + 3x = 5$
inverse of addition
$0 + 3x = 5$
identity of addition
$3x = 5$
multiplication of equality
$(\frac{1}{3})(3)(x) = (\frac{1}{3})(5)$
associative of multiplication
closure of multiplication
$((\frac{1}{3})(3))(x) = (\frac{5}{3})$
inverse of multiplication
$(1)(x) = (\frac{5}{3})$
identity of multiplication
$x = \frac{5}{3}$
No
Does $3(\frac{5}{3}) + 2 = 7$?
Yes
Answer $\frac{5}{3}$
END

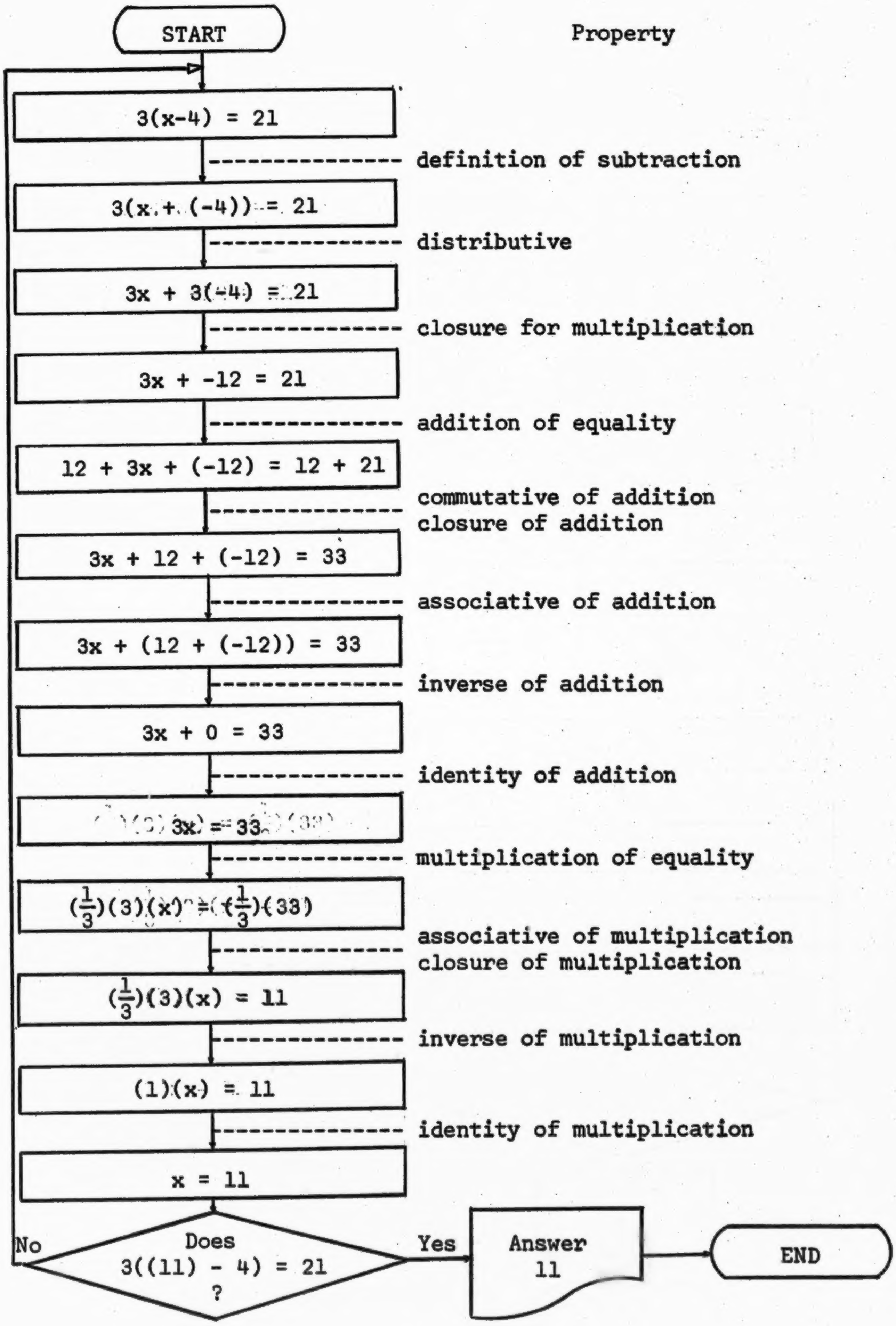
2b)


START
Property
$9 = 5 - \frac{x}{2}$
definition of subtraction
$9 = 5 + (-\frac{x}{2})$
definition of division
$9 = 5 + ((-\frac{1}{2})(x))$
addition of equality
$-5 + 9 = -5 + 5 + ((-\frac{1}{2})(x))$
closure for addition
associative for addition
$4 = (-5+5) + ((-\frac{1}{2})(x))$
inverse for addition
$4 = 0 + (-\frac{1}{2})(x)$
identity for addition
$4 = (-\frac{1}{2})(x)$
multiplication of equality
$(-2)(4) = (-2)(-\frac{1}{2})(x)$
closure for multiplication
associative for multiplication
$-8 = ((-2)(-\frac{1}{2}))(x)$
inverse for multiplication
$-8 = (1)(x)$
identity for multiplication
$-8 = x$
Does $9 = 5 - \frac{(-8)}{2}$?
No
Yes
Answer
-8
END

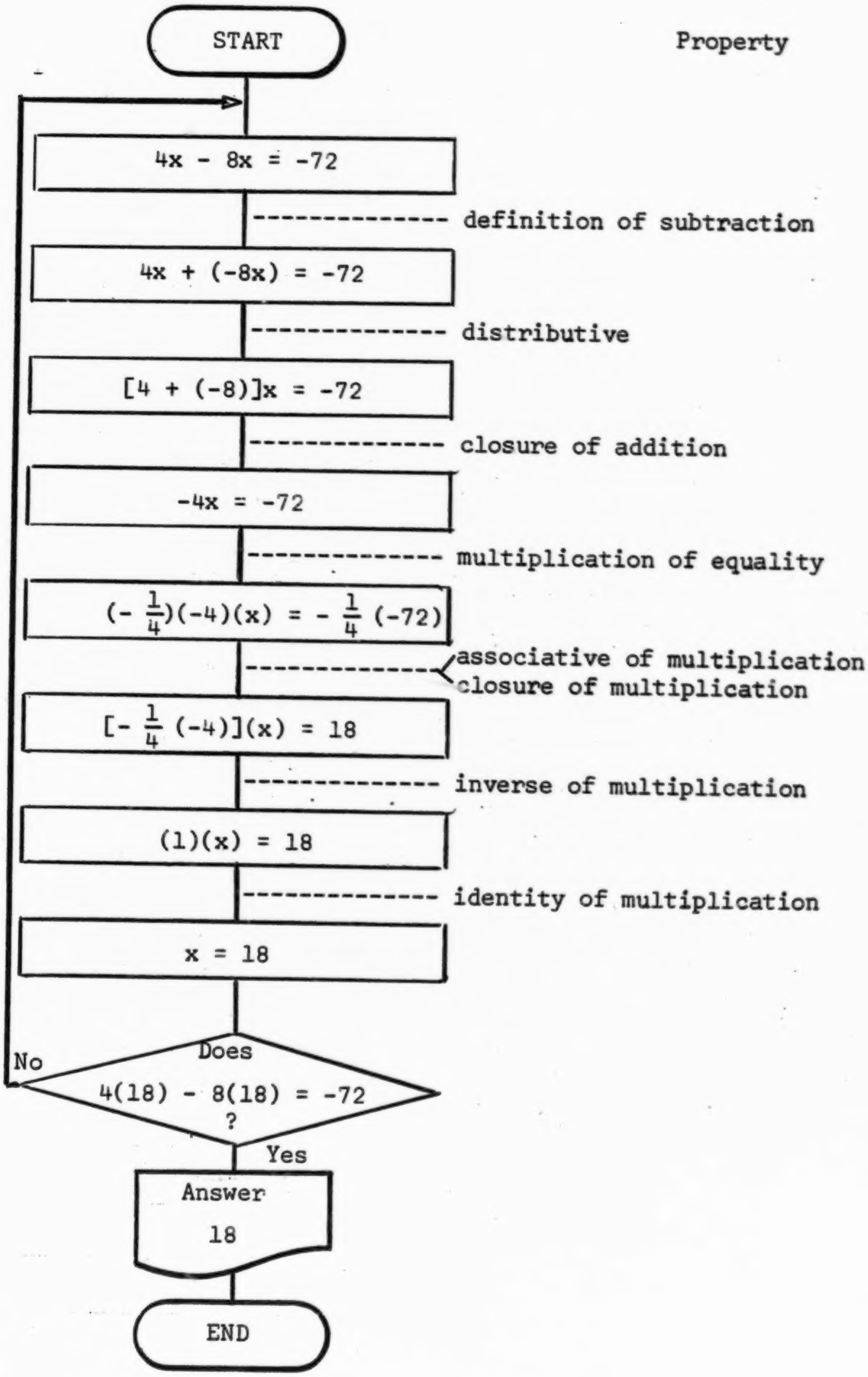
2c)


START
Property
$2\frac{1}{2}x - 17 = 3$
definition of subtraction
$2\frac{1}{2}x + (-17) = 3$
addition of equality
$17 + 2\frac{1}{2}x + (-17) = 17 + 3$
commutative for addition
closure for addition
$2\frac{1}{2}x + 17 + (-17) = 20$
associative for addition
$2\frac{1}{2}x + (17 + (-17)) = 20$
inverse for addition
$2\frac{1}{2}x + 0 = 20$
identity for addition
$2\frac{1}{2}x = 20$
numerals for the same number
$\frac{5}{2}x = 20$
multiplication of equality
$(\frac{2}{5})(\frac{5}{2}x) = (\frac{2}{5})(20)$
associative for multiplication
closure for multiplication
$((\frac{2}{5})(\frac{5}{2}))(x) = 8$
inverse for multiplication
$(1)(x) = 8$
identity for multiplication
$x = 8$
Does $2\frac{1}{2}x - 17 = 3$?
No
Yes
Answer
8
END

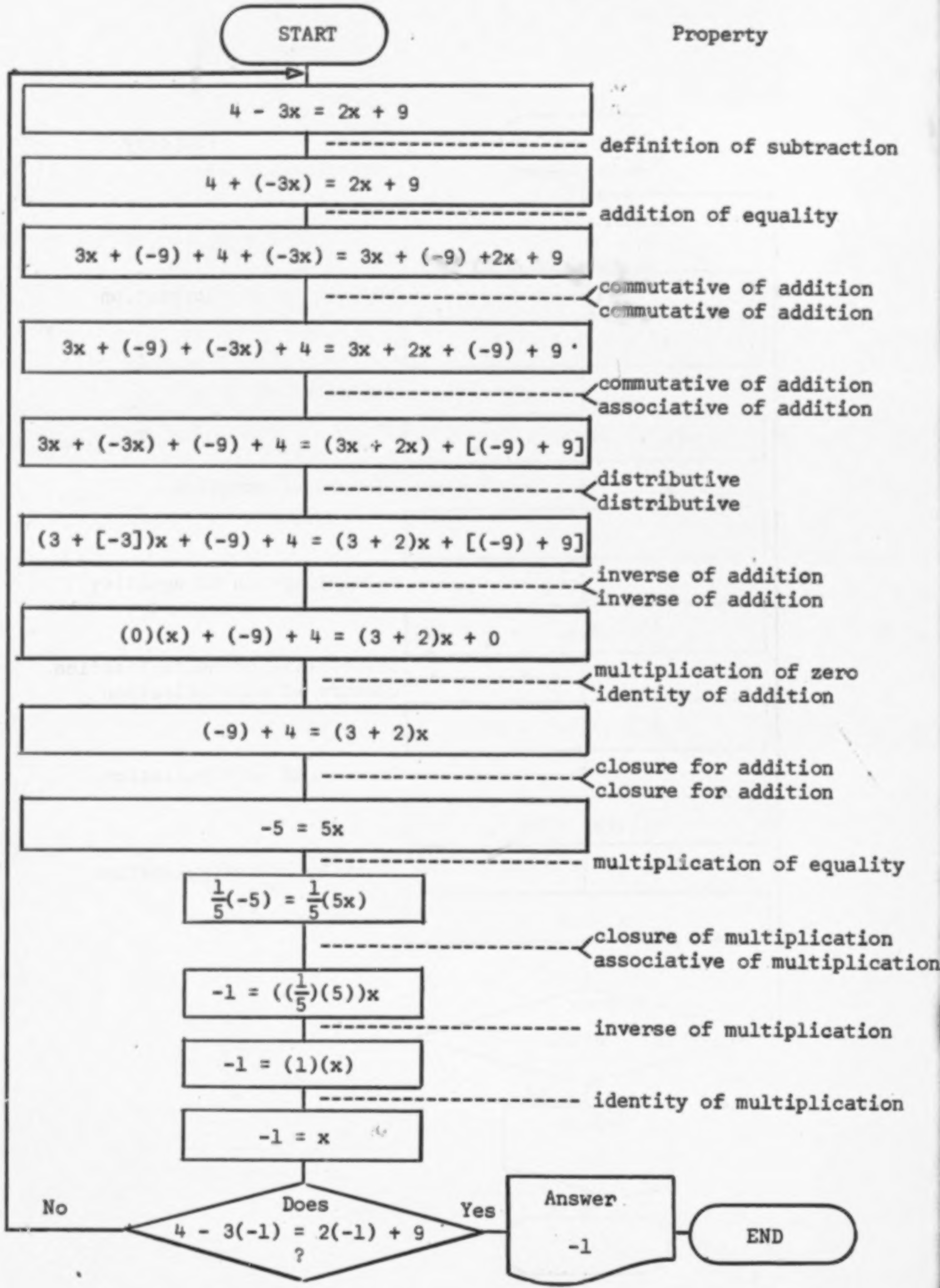
3a)


START
Property
3(x-4) = 21
definition of subtraction
3(x + (-4)) = 21
distributive
3x + 3(-4) = 21
closure for multiplication
3x + -12 = 21
addition of equality
12 + 3x + (-12) = 12 + 21
commutative of addition
closure of addition
3x + 12 + (-12) = 33
associative of addition
3x + (12 + (-12)) = 33
inverse of addition
3x + 0 = 33
identity of addition
3x = 33
multiplication of equality
(1/3)(3)(x) = (1/3)(33)
associative of multiplication
closure of multiplication
(1/3)(3)(x) = 11
inverse of multiplication
(1)(x) = 11
identity of multiplication
x = 11
No
Does 3((11) - 4) = 21 ?
Yes
Answer 11
END

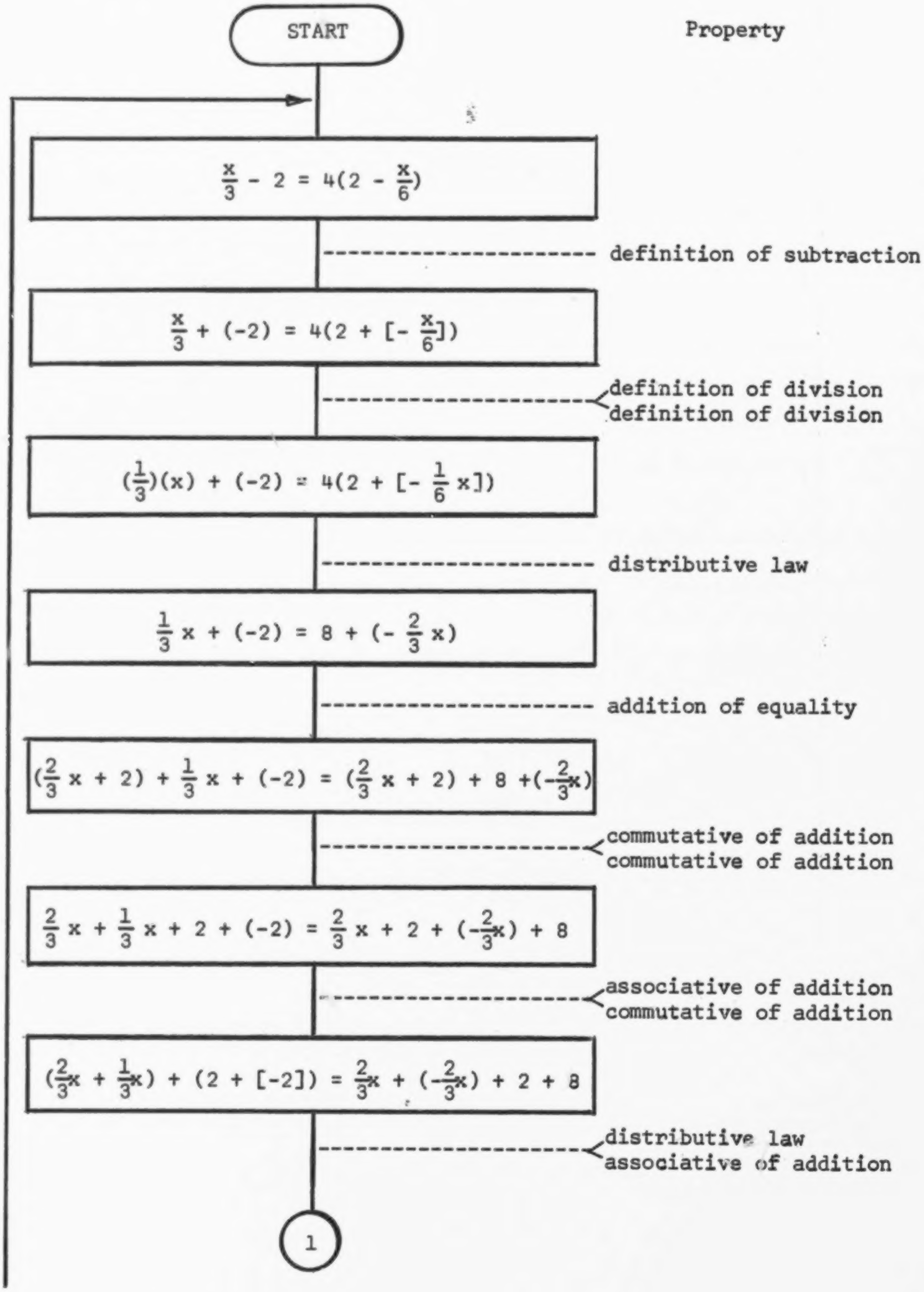
3b)


START
Property
$4x - 8x = -72$
definition of subtraction
$4x + (-8x) = -72$
distributive
$[4 + (-8)]x = -72$
closure of addition
$-4x = -72$
multiplication of equality
$(-\frac{1}{4})(-4)(x) = -\frac{1}{4}(-72)$
associative of multiplication
closure of multiplication
$[-\frac{1}{4}(-4)](x) = 18$
inverse of multiplication
$(1)(x) = 18$
identity of multiplication
$x = 18$
No
Does
$4(18) - 8(18) = -72$
?
Yes
Answer
18
END

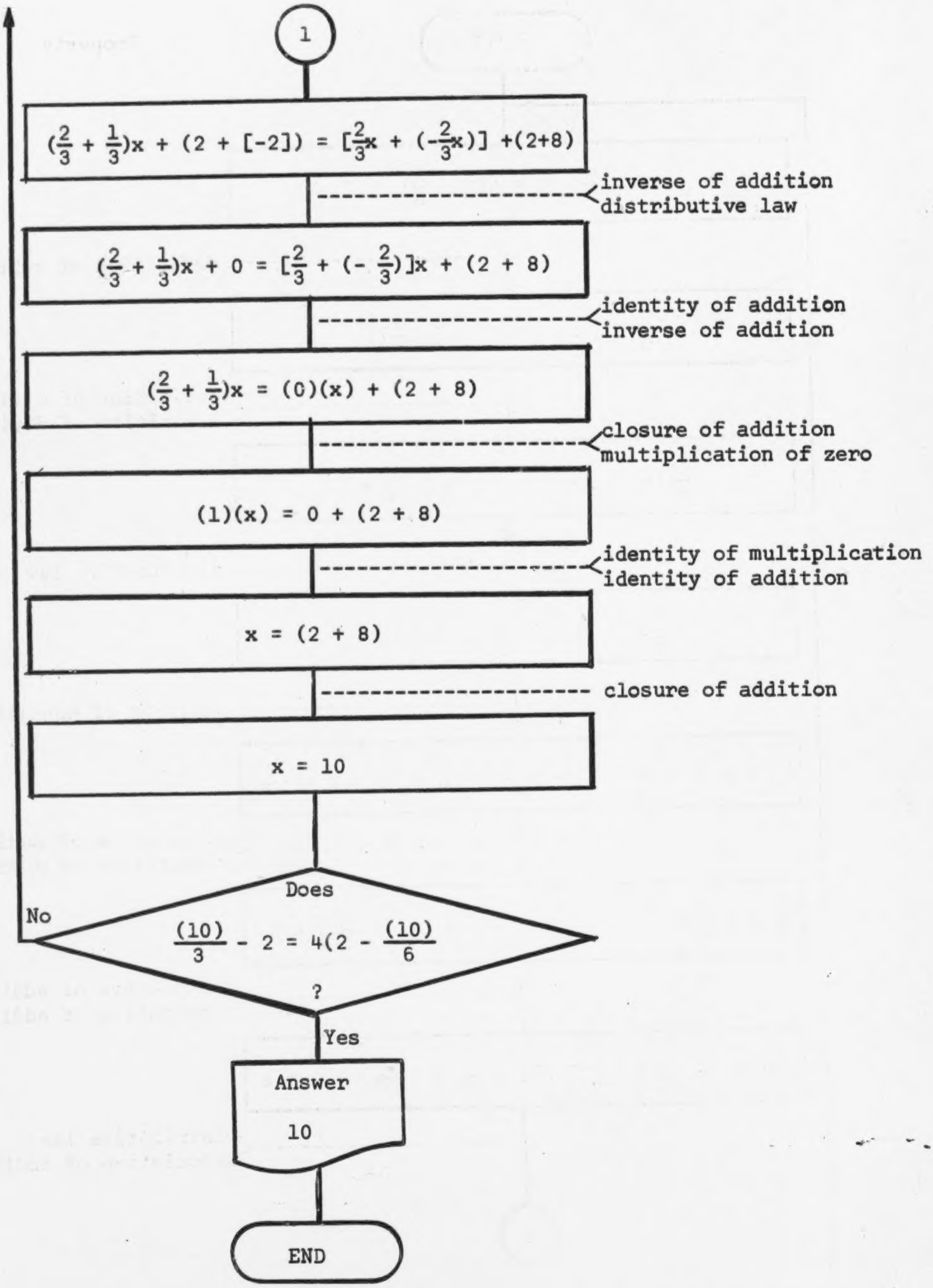
3c)


START
Property
4 - 3x = 2x + 9
definition of subtraction
4 + (-3x) = 2x + 9
addition of equality
3x + (-9) + 4 + (-3x) = 3x + (-9) +2x + 9
commutative of addition
commutative of addition
3x + (-9) + (-3x) + 4 = 3x + 2x + (-9) + 9
commutative of addition
associative of addition
3x + (-3x) + (-9) + 4 = (3x + 2x) + [(-9) + 9]
distributive
distributive
(3 + [-3])x + (-9) + 4 = (3 + 2)x + [(-9) + 9]
inverse of addition
inverse of addition
(0)(x) + (-9) + 4 = (3 + 2)x + 0
multiplication of zero
identity of addition
(-9) + 4 = (3 + 2)x
closure for addition
closure for addition
-5 = 5x
multiplication of equality
$\frac{1}{5}(-5) = \frac{1}{5}(5x)$
closure of multiplication
associative of multiplication
$-1 = ((\frac{1}{5})(5))x$
inverse of multiplication
-1 = (1)(x)
identity of multiplication
-1 = x
Does 4 - 3(-1) = 2(-1) + 9 ?
No
Yes
Answer -1
END

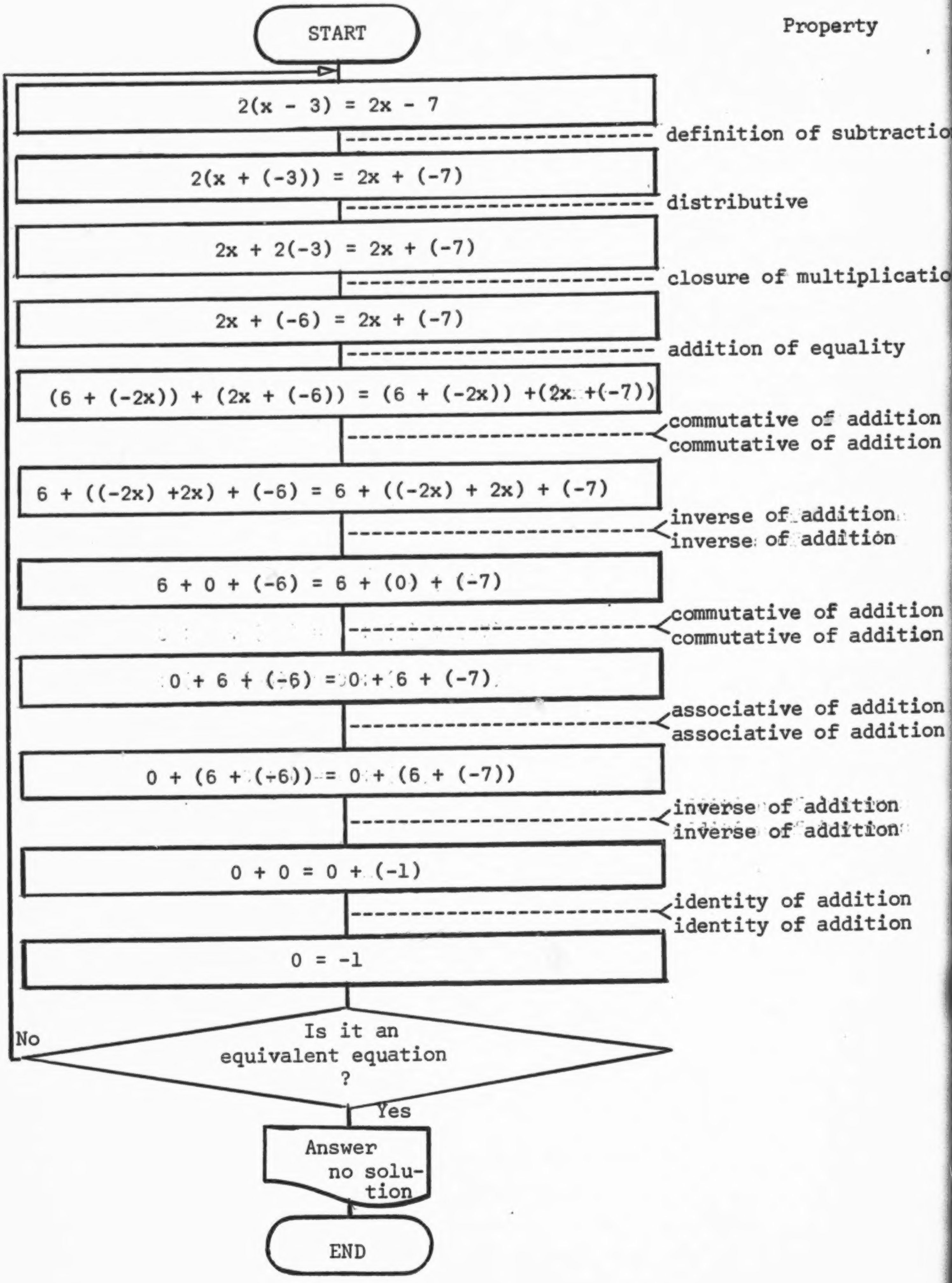
3d)


START
Property
$\frac{x}{3} - 2 = 4(2 - \frac{x}{6})$
definition of subtraction
$\frac{x}{3} + (-2) = 4(2 + [- \frac{x}{6}])$
definition of division
definition of division
$(\frac{1}{3})(x) + (-2) = 4(2 + [- \frac{1}{6} x])$
distributive law
$\frac{1}{3} x + (-2) = 8 + (- \frac{2}{3} x)$
addition of equality
$(\frac{2}{3} x + 2) + \frac{1}{3} x + (-2) = (\frac{2}{3} x + 2) + 8 + (-\frac{2}{3}x)$
commutative of addition
commutative of addition
$\frac{2}{3} x + \frac{1}{3} x + 2 + (-2) = \frac{2}{3} x + 2 + (-\frac{2}{3}x) + 8$
associative of addition
commutative of addition
$(\frac{2}{3}x + \frac{1}{3}x) + (2 + [-2]) = \frac{2}{3}x + (-\frac{2}{3}x) + 2 + 8$
distributive law
associative of addition
1

3d continued)


1
$(\frac{2}{3} + \frac{1}{3})x + (2 + [-2]) = [\frac{2}{3}x + (-\frac{2}{3}x)] + (2+8)$
inverse of addition
distributive law
$(\frac{2}{3} + \frac{1}{3})x + 0 = [\frac{2}{3} + (-\frac{2}{3})]x + (2 + 8)$
identity of addition
inverse of addition
$(\frac{2}{3} + \frac{1}{3})x = (0)(x) + (2 + 8)$
closure of addition
multiplication of zero
$(1)(x) = 0 + (2 + 8)$
identity of multiplication
identity of addition
$x = (2 + 8)$
closure of addition
$x = 10$
Does
$\frac{(10)}{3} - 2 = 4(2 - \frac{(10)}{6}$
?
No
Yes
Answer
10
END

3e)


START
Property
2(x - 3) = 2x - 7
definition of subtractio
2(x + (-3)) = 2x + (-7)
distributive
2x + 2(-3) = 2x + (-7)
closure of multiplicatio
2x + (-6) = 2x + (-7)
addition of equality
(6 + (-2x)) + (2x + (-6)) = (6 + (-2x)) +(2x +(-7))
commutative of addition
commutative of addition
6 + ((-2x) +2x) + (-6) = 6 + ((-2x) + 2x) + (-7)
inverse of addition
inverse of addition
6 + 0 + (-6) = 6 + (0) + (-7)
commutative of addition
commutative of addition
0 + 6 + (-6) = 0 + 6 + (-7)
associative of addition
associative of addition
0 + (6 + (-6)) = 0 + (6 + (-7))
inverse of addition
inverse of addition
0 + 0 = 0 + (-1)
identity of addition
identity of addition
0 = -1
Is it an equivalent equation ?
No
Yes
Answer no solution
END

PRIME NUMBERS AND "BASIC" COMPUTER LANGUAGE

by Ray Platt and Ken Marine

Introduction

This is a project for one or two class periods in a Junior or Senior High mathematics course. The purpose is to show that a computer can help in determining prime numbers. The individual students should know the terms square root, natural numbers, factor, divisor, multiple, even and odd.

The computer approach will be incorporated with the prime numbers topic used in most mathematics text books. Two flow charts and two programs are given for each section of this topic. The students might generate others that fit their individual program or procedures.

This project consists of three parts. Part I is a brute force attack on a number. Part II is a more sophisticated method of determining if a number is prime. Part III is a section to list prime numbers with the use of the Sieve of Eratosthenes.

Definition

A natural number with only two different factors is called a prime number.

PART I

Brute Force

First ask the students how they can tell if a number is prime. Give them some numbers such as 1, 2, 4, 7, 9, and 15. Then give them a large number such as 529 which seems to be prime but is devisible by 23. Then ask the students how they worked on the numbers given. This should lead to a flow chart such as Flow Chart 1 or 2. In this attack, a number is divided by every number starting with 2 up to the given number. Program 1 prints out which numbers are prime and which ones are not and lists two possible factors. Program 2 prints out factors of the given number other than one and the number itself if such factors exist. If there are no such factors, it prints out one and the number itself. Inspection of the results reveals which numbers are prime. Students of your class might be asked to compare the run time of this type of program with a more refined program of Part II. Note: Try a large prime such as 2351.

PART II

Division by Numbers Less than or Equal to the Square Root of a Given Number

After the students start dividing by all the numbers less than the number in question, you can ask them to find a shorter method. Take a number such as 53. Play computer using one of the programs given in Part I.

This might lead to the idea that a limit of the square root of the given number can be set on the divisor. (See Flow Charts 3 and 4). The programs given here show two ways in which the square root step is used. One program starts with the square root and works down to one. The other works from 2 as the first divisor. (See Programs 3 and 4). Note: The run time is the same for Part I and Part II programs. Try other numbers to make a good comparison.

PART III

The Sieve of Eratosthenes

The Sieve of Eratosthenes is commonly introduced by having the students make a chart (Figure A) which lists the numbers from 1 to 100. First the number one is crossed out, because it does not fit the definition. Then 2 is circled as a prime. Then each multiple of 2 is crossed out. Then 3 is circled, and all of the multiples of 3 are crossed out. This process is used until the numbers that are circled are considered to be prime. This idea leads directly to the flow charts for this section (5 and 6). In each case the only divisors used are prime numbers. In Program 5 the square root limit is used. Note also the different methods used for storing the primes and printing the primes. (See Programs 5 and 6). It must be mentioned that the dimension of the total of primes (see step one of the programs) is a point for the students to determine. You also may wish to print out the total at the end of your list of primes.

PROGRAM 1

```
10 READ M
20 LET N=2
30 LET K=INT(M/N)
40 LET L=M-K*N
50 IF L<>0 THEN 80
60 PRINT M,"IS NØT PRIME;FACTORS ARE",K,N
70 GØ TØ 10
80 LET N=N+1
90 IF N<M THEN 30
100 PRINT M,"IS PRIME"
110 GØ TØ 10
120 DATA 31,121,123,113
130 END
RUN
WAIT.

M1        15:01    PX THU 08/17/67

 31           IS PRIME
 121          IS NØT PRIME;FACTØRS ARE      11          11
 123          IS NØT PRIME;FACTØRS ARE      41          3
 113          IS PRIME
```

```
ØUT ØF DATA IN  10

TIME:     0 SECS.
```

PROGRAM 2

```
10 READ N
20 LET D=2
30 LET Q=N/D
40 IF INT(Q)*D=N THEN 70
50 LET D=D+1
60 GØ TØ 30
70 PRINT N;"=";Q;"*";D
80 GØ TØ 10
90 DATA 31,121,123,113
100 END
KEY
READY.

RUN
WAIT.

P1         14:49     PX THU 08/17/67

 31    = 1    * 31
 121   = 11   * 11
 123   = 41   * 3
 113   = 1    * 113

ØUT ØF DATA IN  10

TIME:   0 SECS.
```

PROGRAM 3

```
10 READ M
20 IF M<2 THEN 130
30 LET N=2
40 LET K=INT(M/N)
50 LET L=M-K*N
60 IF L<>0 THEN 90
70 PRINT M,"ENTER ONLY INTEGERS >=2"
80 GØ TØ 10
90 LET N=N+1
100 IF N<=SQR(M) THEN 40
110 PRINT M,"IS PRIME"
120 GØ TØ 10
130 PRINT M,"IS WRØNG"
140 GØ TØ 10
150 DATA 31,121,123,113
160 END

RUN
WAIT.

L          15:03   PX THU 08/17/67
```

```
 31            IS PRIME
 121           IS NØT PRIME
 123           IS NØT PRIME
 113           IS PRIME

ØUT ØF DATA IN  10

TIME:   0 SECS.
```

PROGRAM 4

```
10 READ N
20 LET D=INT(SQR(N))
30 IF INT(N/D)*D=N THEN 90
40 LET D=D-1
50 IF D=1 THEN 70
60 GØ TØ 30
70 PRINT N;"IS PRIME"
80 GØ TØ 10
90 PRINT N;"IS NØT PRIME"
100 PRINT D;"*";INT(N/D);"=";N
110 GØ TØ 10
120 DATA 31,121,123,113
130 END

RUN

P3        14:59   PX THU 08/17/67

 31   IS PRIME
 121  IS NØT PRIME
 11   * 11   = 121
 123  IS NØT PRIME
 3    * 41   = 123
 113  IS PRIME

ØUT ØF DATA IN  10

TIME:   0 SECS.
```

PROGRAM 5

```
10 DIM A(50)
20 LET A(1)=1
25LET A(2)=2
30 FØR M=3 TØ 99
40 FØR N=2 TØ SQR(M)
50 IF A(N)=0 THEN 100
60 LET K=INT(M/A(N))
70 LET L=M-K*A(N)
80 IF L=0 THEN 150
90 NEXT N
100 FØR I=1 TØ 50
110 IF A(I)<>0 THEN 140
120 LET A(I)=M
130 GØ TØ 150
140 NEXT I
150 NEXT M
```

```
160 FØR I=2 TØ 50
170 IF A(I)=0 THEN 200
180 PRINT A(I);
190 NEXT I
200 END

RUN

M         15:10    PX THU 08/17/67

 2     3     5     7     11    13    17    19    23    29    31
 37    41    43    47    53    59    61    67    71    73    79
 83    89    97

TIME:   2 SECS.
```

PROGRAM 6

```
05 DIM X(50)
10LET Z=1
20 READ N
30 FØR A=3 TØ N
40 FØR M=1 TØ Z
50 LET X(1)=2
60 LET Q=INT(A/X(M))
70 IF A-(X(M)*Q)=0 THEN 110
80 NEXT M
90 LET Z=Z+1
100 LET X(Z)=A
110  NEXT A
120 FØR M=1 TØ Z
130 PRINT X(M);
140 NEXT M
150 DATA 100
160 END

RUN

P2      14:56    PX THU 08/17/67

 2     3     5     7     11    13    17    19    23    29    31
 37    41    43    47    53    59    61    67    71    73    79
 83    89    97

TIME:  2 SECS.
```

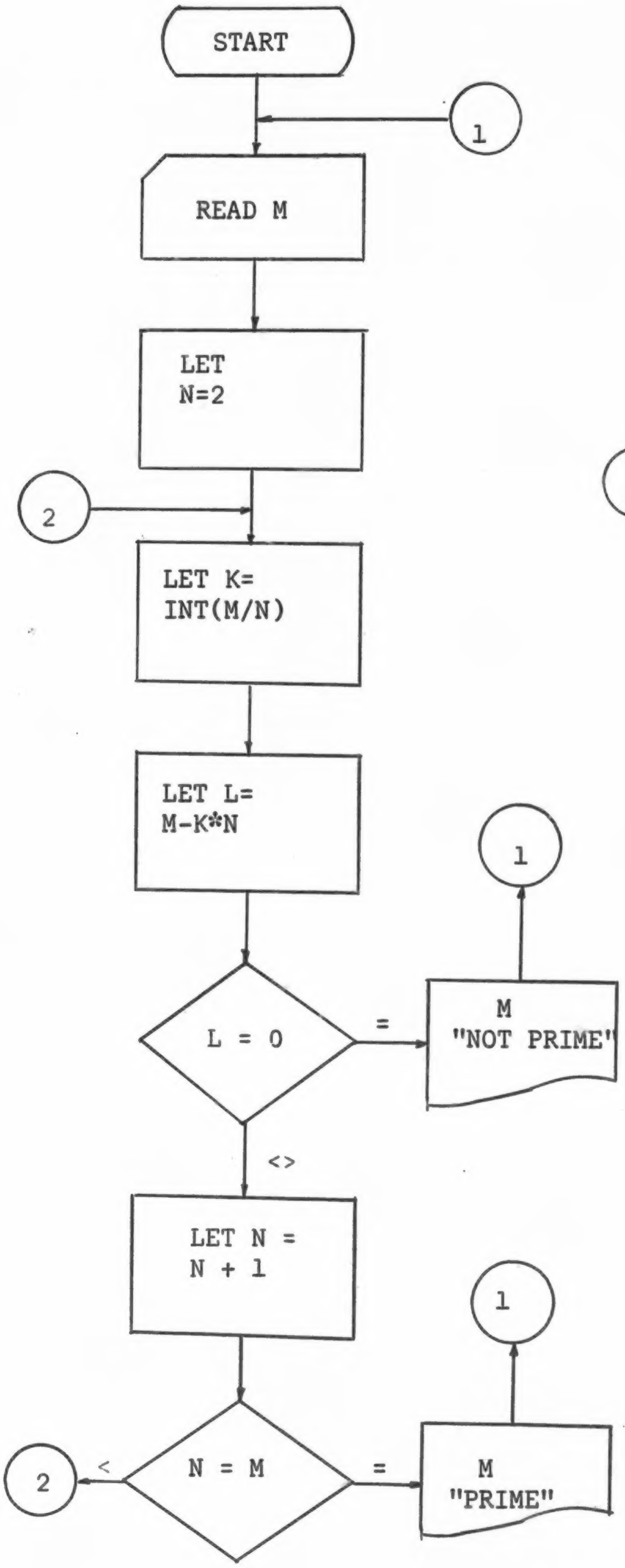
Summary

Those programs and charts are guides in the construction of material for determining prime numbers. In each case refinements can be made on these programs. During the writing of this project, tests were made on larger numbers. It was found that the run time can be cut by several seconds if only odd numbers are considered for primes. This step can easily be inserted with a Step Statement of 2.

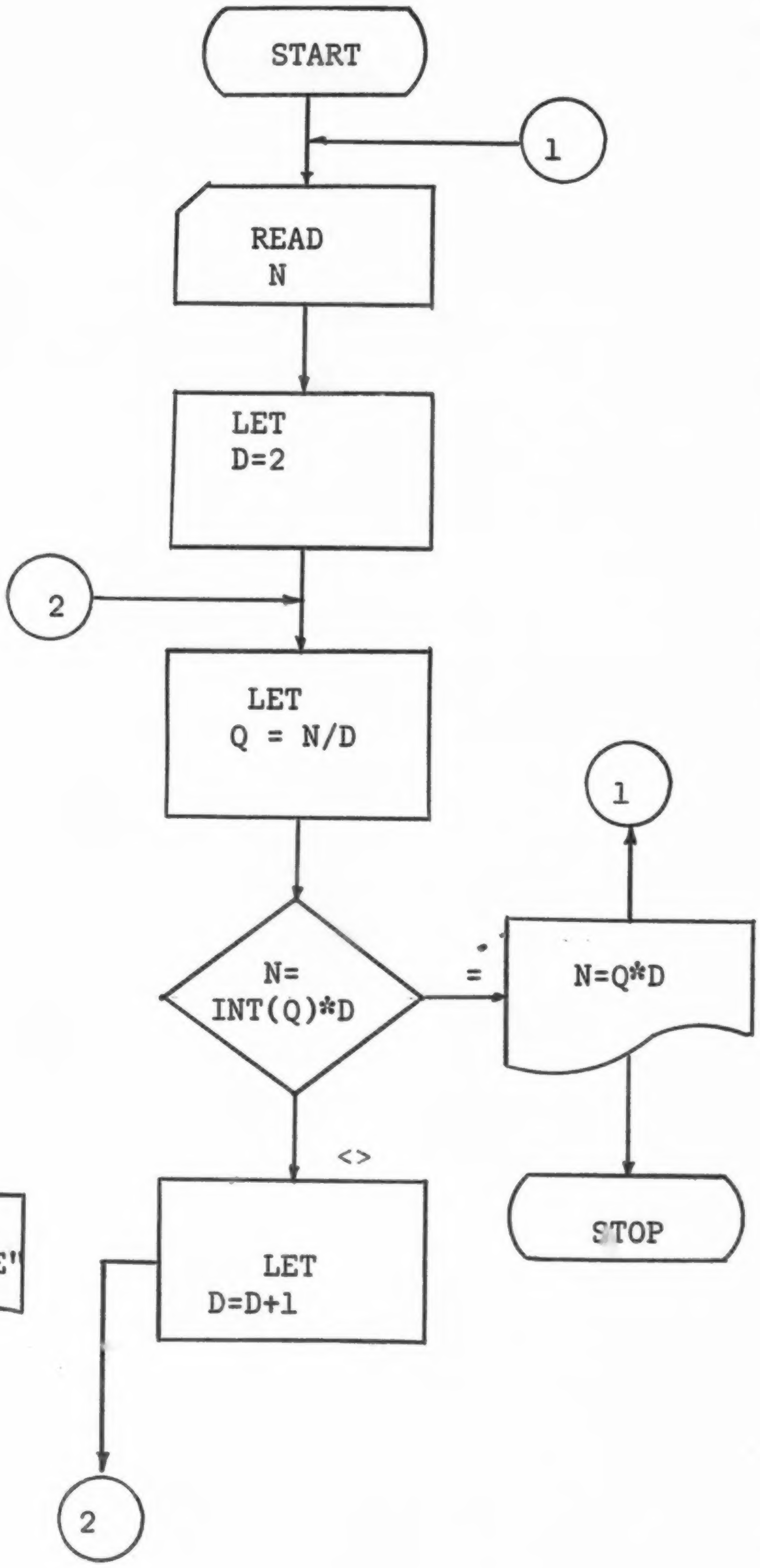
Each section of this project can be used separately if so desired. All three sections will work in connection with most mathematics texts which cover prime numbers.

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

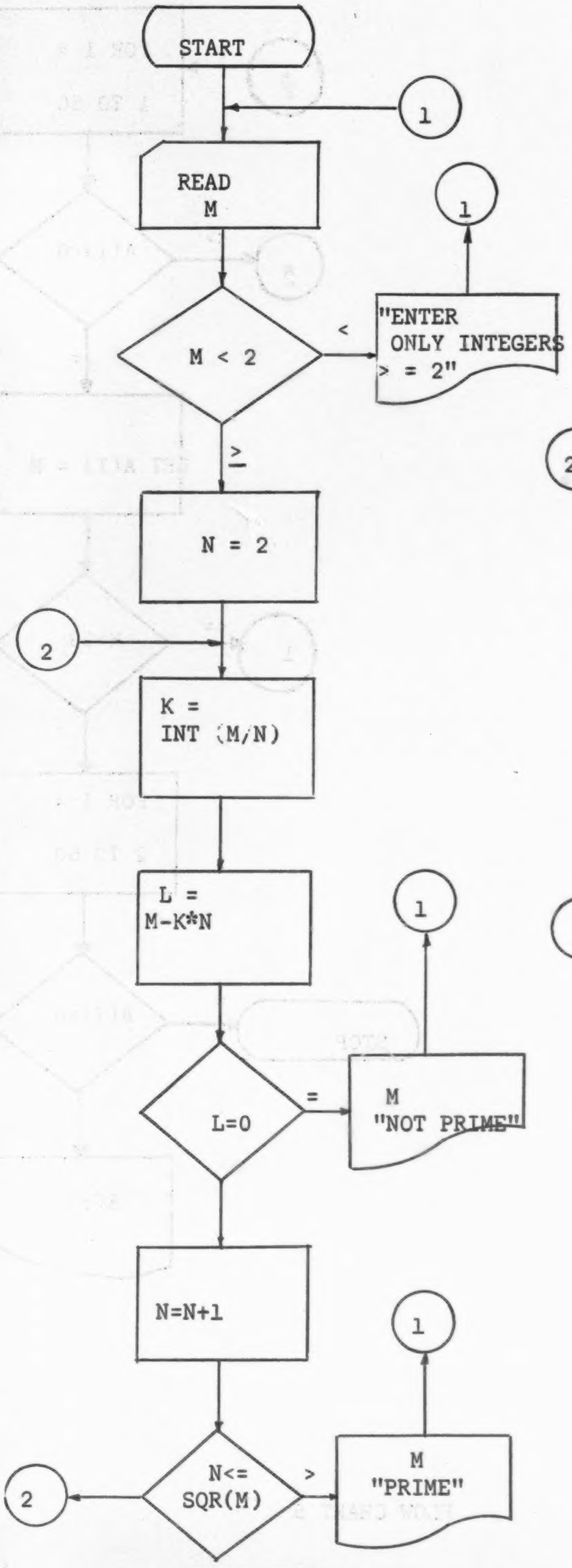
FIGURE A


START
1
READ M
LET
N=2
2
LET K=
INT(M/N)
LET L=
M-K*N
1
L = 0
=
M
"NOT PRIME"
<>
LET N =
N + 1
1
2
<
N = M
=
M
"PRIME"

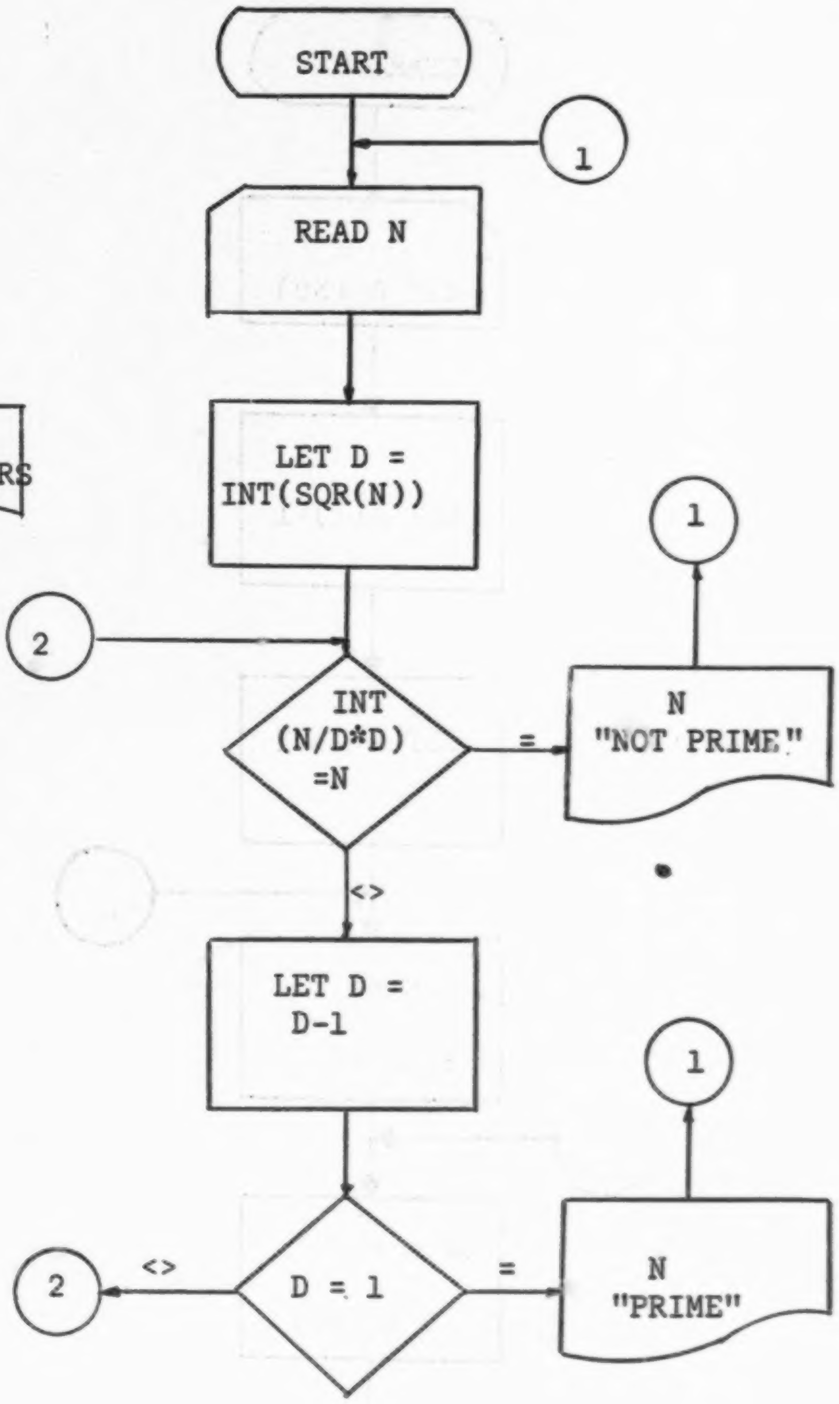
FLOW CHART 1


START
1
READ
N
LET
D=2
2
LET
Q = N/D
1
N=
INT(Q)*D
=
N=Q*D
<>
LET
D=D+1
STOP
2

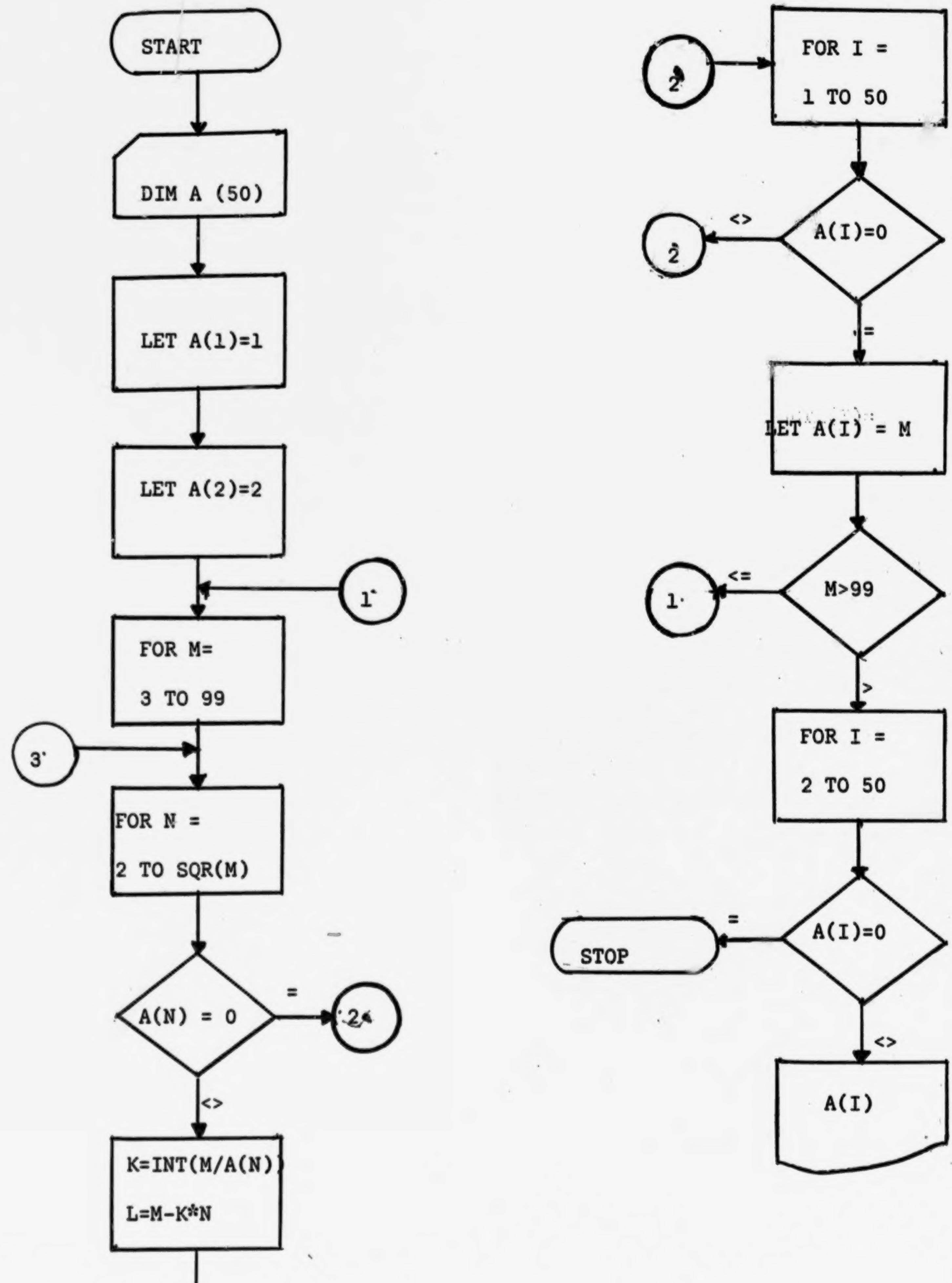
FLOW CHART 2


START
1
READ
M
1
M < 2
<
"ENTER
ONLY INTEGERS
> = 2"
≥
N = 2
2
K =
INT (M/N)
L =
M-K*N
1
L=0
=
M
"NOT PRIME"
N=N+1
1
2
N<=
SQR(M)
>
M
"PRIME"

FLOW CHART 3

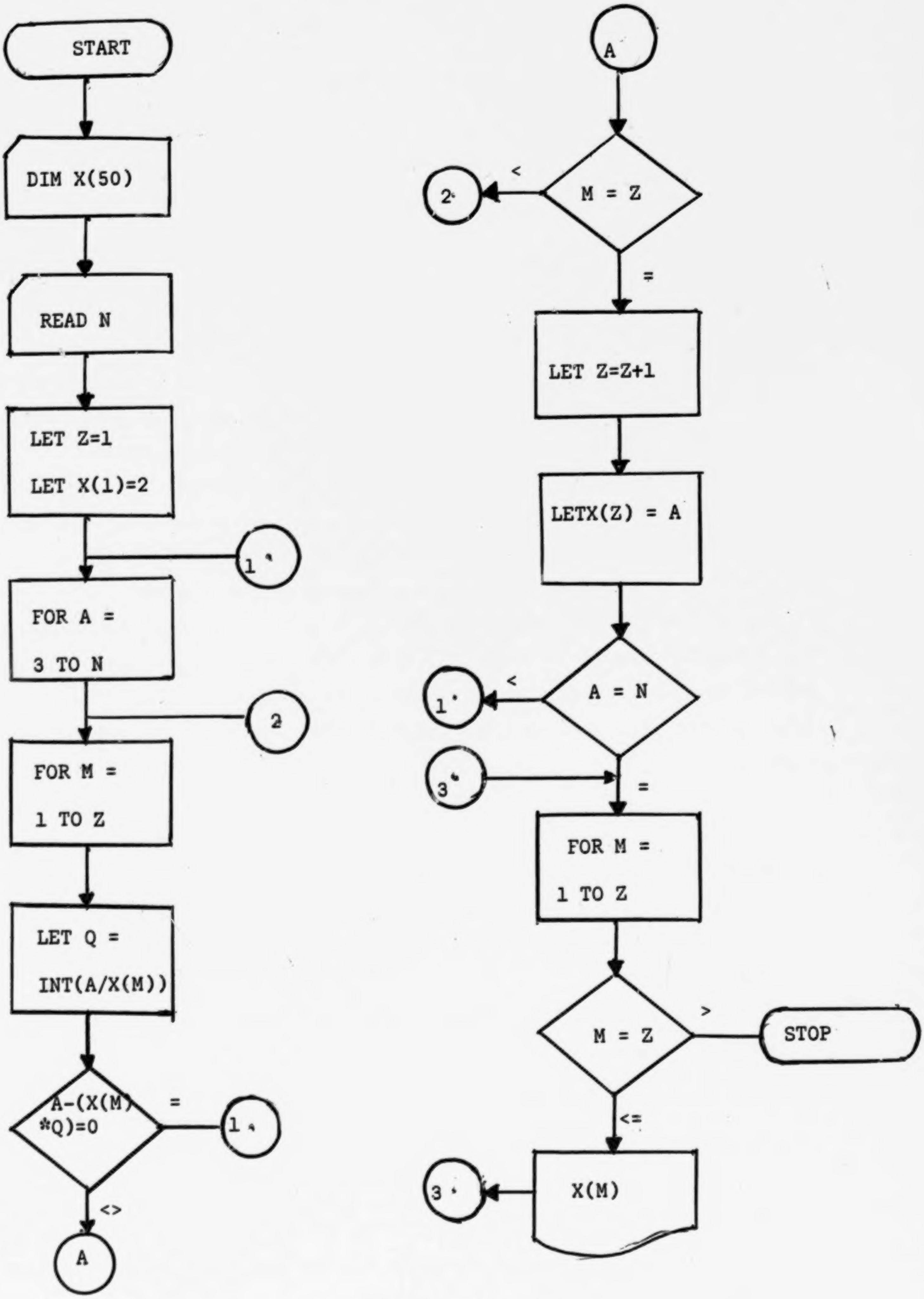

START
1
READ N
LET D =
INT(SQR(N))
1
2
INT
(N/D*D)
=N
=
N
"NOT PRIME"
<>
LET D =
D-1
1
2
<>
D = 1
=
N
"PRIME"

FLOW CHART 4


START
DIM A (50)
LET A(1)=1
LET A(2)=2
1
FOR M=
3 TO 99
3
FOR N =
2 TO SQR(M)
A(N) = 0
=
2
<>
K=INT(M/A(N))
L=M-K*N
<>
3
L = 0
=
1
2
FOR I =
1 TO 50
2
<>
A(I)=0
=
LET A(I) = M
1
<=
M>99
>
FOR I =
2 TO 50
STOP
=
A(I)=0
<>
A(I)

FLOW CHART 5

④


START
DIM X(50)
READ N
LET Z=1
LET X(1)=2
1
FOR A =
3 TO N
2
FOR M =
1 TO Z
LET Q =
INT(A/X(M))
A-(X(M)
*Q)=0
=
1
<>
A
A
<
2
M = Z
=
LET Z=Z+1
LETX(Z) = A
<
1
A = N
3
=
FOR M =
1 TO Z
>
M = Z
STOP
<=
3
X(M)

FLOW CHART 6

AN APPROACH TO THE INTRODUCTION OF COMPUTER PROGRAMMING IN JUNIOR HIGH SCHOOL MATHEMATICS

by Helen E. Schweizer

The purpose of this project is to develop materials for a junior high school student which would reinforce and extend his concepts of number and number operations and help him to develop a familiarity with computer programming.

It is assumed that the student has identified and operated with the sets of natural, whole, and non-negative rational numbers (both common and decimal fractions) and has an awareness of negative numbers; that he has had experience working with several number bases -- place value, symbols, conversion from one base to another; and that he has made limited use of an exponent. Primarily, the number operations which he would have used are addition, subtraction, multiplication, and division.

The study guides which follow begin with a mathematical definition appropriate for this level, or with a suitable concept, and attempt to relate this to computer techniques. They are intended as curriculum guidelines and, in general, might be used as a basis for class discussion or for individual study by a student. In some instances, an attempt has been made to provide student material and problem exercises, but the knowledgeable teacher will realize that these need to be augmented with appropriate class work.

Study Guide I

Number

1. What is the first counting (natural) number? Write the second counting number. How could it be obtained from the first number? Write the third counting number. How could it be obtained from the second?

   How can each counting number be obtained from the previous counting number?

   What is the largest counting number you know?

   (Teacher: Do not accept "infinity." By showing that infinity does not obey our rules for number, demonstrate to the student that infinity is not a number.)

2. The set of whole numbers contains all the counting numbers and one other number. What is this number?

Computer

With certain commands the computer can produce those elements of the set of counting numbers or the set of whole numbers which are specified. This is the program for the first 100 whole numbers.

```
1 LET X = 0
2 PRINT X
3 LET X = X + 1
4 IF X < = 99 THEN 2
5 END
```

The following comments explain the above program in detail.

1. Each line begins with a number.
2. The sequence of the line numbers indicates to the computer the order of steps to follow.
3. "X" is a variable. A variable in BASIC may be indicated by any letter. A variable stands for a number.
4. "LET" is one of the statements used to assign a value to a variable; it is not used to state an equality. The value of the variable does not change until the next LET statement is encountered with a value for that variable.
5. "PRINT X" tells the computer to print each value for X.
6. The "IF"--"THEN" statement places a limit on the size of the set.
7. "END" tells the computer to stop reading the program. Without this command the computer looks for more information.
8. The computer would perform with the above program as follows:
   1) Let X = 0 because of line 1.
   2) Print 0 because of line 2.
   3) Change the value of X to 0 + 1 (or 1) because of line 3.
   4) Determine whether the latest value for X would be less than or equal to 99 because of line 4.
   5) Because the value of X would be less than 99, return to line 2 and print 1.
   6) The present value of X would be increased by 1 because of line 3. The new value of X would be 2.
   7) Continue to run through the loop of lines 2, 3, and 4 until X would not be less than or equal to 99.
   8) At this time continue to the next line, line 5, and stop.

Computer Activity I

A teacher may be able to create interest and promote student involvement by using students to simulate a computer program. Some students, particularly those who clearly understand the computer program for whole numbers, could function as programmers. The other students in the class could respond to the commands of the program.

A group of six students with the assignment of a computer program for the first 15 counting numbers might perform as follows:

1. One student would write a complete program similar to that for the set of whole numbers.
2. Each of the other 5 students would be assigned a different line of the program; his identification would be the number of his line. He would determine his function in the assigned program.
3. The programmer would check individually with each of the 5 students to be certain each had the correct command and understood his responsibility with it.
4. Lines #1 and #3 would be given blank cards. Each card would be used to complete an operation within the computer.
5. Line #2 would be at the blackboard to print the results requested by the program.

This is the program:

```
1  LET X = 1
2  PRINT X
3  LET X = X + 1
4  IF X < = 15 THEN 2
5  END
```

The program might run as follows:

1. The programmer would instruct the program to "start."
2. #1 would write "1" on a card and would give the card to #2.
3. #2 would print "1" on the blackboard and would give the card to #3.
4. #3 would keep that card. On a new card he would write "2", the result of adding 1 to the previous value of X. He would give the card to #4.
5. #4 would determine that 2 is less than or equal to 15. He would give the card to #2.
6. The loop would continue to repeat until X > 15. Then #4 would give the card to #5.
7. The computer would stop.

Upon completion of the program, each student would make a diagram (flow chart) to show the purpose of each command and its location in the total program. A discussion of these flow charts should lead to the recognition of the need for uniformity of symbolism and to an agreement on the symbols to be used in the flow charts for start, input, instructions, decisions, output, and stop.

Many variations of this activity are possible. The responsibilities within a program may be rotated. A related mathematical concept or a completely different concept may serve as the basis for developing a new program. As new computer commands are learned, attention may be given to the various program possibilities which are available for a given program. In any case, the complexity of the activity should be dictated by the ability of the group involved.

The above activity might also be utilized to develop a greater student appreciation of the time-saving aspect of the computer. A comparison can be made between the running times of the class and the computer for a particular program. Actual computing time can also be compared.

Computer Activity II

1. Write the first 15 even counting numbers.
2. How is the second even counting number obtained from the first?
3. How is each successive even counting number obtained from the previous even counting number?
4. How could the program for the whole numbers be changed so that even counting numbers could be obtained?
5. Write a program for the even counting numbers between 117 and 202.
6. Should any changes be made in the program in #5 to obtain the odd counting numbers between 117 and 202? Explain your answer.

Computer Activity III

1. Write five different multiples of three. What characteristic is common to these five numbers?
2. Write the first five multiples of three which are larger than 39.
3. Begin with 40, and count by 3's to 60. Are these numbers multiples of 3? Explain your answer.
4. Write a program for consecutive multiples of 3.
5. Write a program which will obtain the numbers for #3.

Study Guide II

Number

In base 10 the counting and whole numbers may be written by using 10 different symbols which are called digits. These digits are 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. The position of these digits in the numeral determines their value. The place values of the first six places to the left of the decimal point in base 10 may be expressed as follows:

100000	10000	1000	100	10	1
10^5	10^4	10^3	10^2	10^1	10^0

The value of 29056 could be expressed as:

$2(10000) + 9(1000) + 0(100) + 5(10) + 6(1)$

or as:

$2(10^4) + 9(10^3) + 0(10^2) + 5(10^1) + 6(10^0)$

In base 8 the counting and whole numbers may be written by using eight different symbols (0, 1, 2, 3, 4, 5, 6, 7) and the place values may be expressed as powers of eight. The place values of the first six places in base eight are as follows:

base 8:	100000	10000	1000	100	10	1
base 8:	10^5	10^4	10^3	10^2	10^1	10^0
base 10:	32768	4096	512	64	8	1
base 10:	8^5	8^4	8^3	8^2	8^1	8^0

The value of 2314_8 may be expressed in base 8 as follows:

$2(10^3) + 3(10^2) + 1(10^1) + 4(10^0)$

or in base 10 as:

$2(8^3) + 3(8^2) + 1(8^1) + 4(8^0)$

$2(512) + 3(64) + 1(8) + 4(1)$

$1024 + 192 + 8 + 4$

1228

In base 2 the counting and whole numbers may be written by using 2 different symbols (0, 1) and the place values may be expressed as powers of two. The place values of the first six places in base two may be expressed as follows:

base 2:	100000	10000	1000	100	10	1
base 2:	10^{101}	10^{100}	10^{11}	10^{10}	10^1	10^0
base 10:	32	16	8	4	2	1
base 10:	2^5	2^4	2^3	2^2	2^1	2^0

The value of 111001_2 may be expressed in base 2 as follows:

$1(10^{101}) + 1(10^{100}) + 1(10^{11}) + 0(10^{10}) + \ (10^{1}) + 1(10^{0})$

or in base 10 as:

$1(2^5) + 1(2^4) + 1(2^3) + 0(2^2) + 0(2^1) + 1(2^0)$

$1(32) + 1(16) + 1(8) + 0(4) + 0(2) + 1(1)$

$32 + 16 + 8 + 0 + 0 + 1$

57

Computer

The computer places the following restrictions on the numerals which it will accept:

1. All numerals should be base 10 numerals. If they are not, they are read as if they were in base 10. Knowledge of base 2 and base 8 is necessary for understanding the internal operations of the computer.
2. Numerals which contain commas are not accepted.
3. The maximum number of digits which will be accepted varies with the computer. Assume that the maximum number of digits accepted by this computer is six. If a numeral contains more than 6 digits, it may be rewritten with 6 or less digits and a power of 10.

$$\begin{aligned}4650000 &= 465000(10^1)\\ &= 46500(10^2)\\ &= 4650(10^3)\\ &= 465(10^4)\\ &= 46.5(10^5)\\ &= 4.65(10^6)\\ &= .465(10^7)\\ &= .0465(10^8)\end{aligned}$$

The computer will accept the powers of 10 only if they are indicated with the letter E followed by the exponent. The above would be rewritten as follows for computer use:

$465000(10^1)$	465000E1
$46500(10^2)$	46500E2
$4650(10^3)$	4650E3
$465(10^4)$	465E4
$46.5(10^5)$	46.5E5
$4.65(10^6)$	4.65E6
$.465(10^7)$	.465E7
$.0465(10^8)$	.0465E8

The letter E must always be preceded by at least one digit.

Computer Activity I

Play computer!

Which of the following will you accept?

1. 9405
2. 32410_8
3. 208,507
4. 1059000
5. 101100_2
6. 7690000000
7. 2317_8
8. 60781
9. 1001101_2
10. 32000000
11. 562_8
12. 10000000000

Rewrite the numerals which were rejected in a form which will be accepted.

Study Guide III

Number Operations

1. List the operations which may be performed with counting numbers.
2. Given the counting number A and the counting number B, will the answers to each of the following problems always be a counting number?

   A + B

   A - B

   A x B

   A ÷ B

   A^B
3. If the answer to any of the above is no, write the conditions which would be necessary to obtain an answer which is a counting number.

Computer Activity I

Make 5 copies of the following program:

```
1  READ A, B
2  PRINT A, B, A___B
3  DATA 12, 2, 15, 3, 10, 5
4  DATA 8, 4, 2, 5, 9, 2
5  GO TO 1
6  END
```

1. Replace the blank separating A and B in line 2 in each of the programs with a different one of the following number operation symbols:

*, -, +, /, ↑

2. Examine the results which are printed by the computer to determine the operation which is indicated by each of the symbols. Write each symbol with the name of the operation that it indicates.
3. Are the results for each operation counting numbers? Do these results verify your earlier conclusions concerning this?

Interpretation of the above program:

1. "READ A, B" identifies the variables with which the computer will operate. The variables are separated with commas.
2. The READ statement must always be followed later in the program with a DATA statement. The DATA statement gives the values of the variables in the order in which they are listed in the READ statement and separates them with commas.
3. In this program the operation to be performed on the variables A and B is indicated in the PRINT statement. The computer will print the values of A, B, and A___B.
4. After the computer has used the 12, 2 values for A, B, it will go to line 5 which tells it to GO TO 1. The READ statement tells it to look for new data so 15, 3 are used as the new values for A, B. A, B, A___B are printed. The computer will continue to work through the loop of lines 1, 2, 3, 4, 5. When there is no more DATA in line 3, it will read the DATA in line 4. When there is no more DATA in line 4, it will print "OUT OF DATA IN 4" and stop.

Computer Activity II

1. Solve the following problems by computer.
    1) 5*2↑4
    2) 5↑2*4
    3) 8/2↑2
    4) 8↑2/2
    5) 40*10/5
    6) 40/10*5
    7) 11-2↑3
    8) 11↑2-3
    9) 90-15/5
    10) 90/15-5
    11) 23-7*3
    12) 23*7-3
    13) 4+2↑3
    14) 4↑2+3
    15) 16+8/2
    16) 16/8+2
    17) 3+2*5
    18) 3*2+5
    19) 6+4-3
    20) 6-4+3
2. By examination of the answer of each of the above problems, determine which operation was performed first in each problem. Circle the symbol for that operation.
3. What order do you think the computer would follow in solving a problem in which all five operations were involved?
4. Write a problem which involves all five operations. Solve this problem by computer to test your answer to #3.

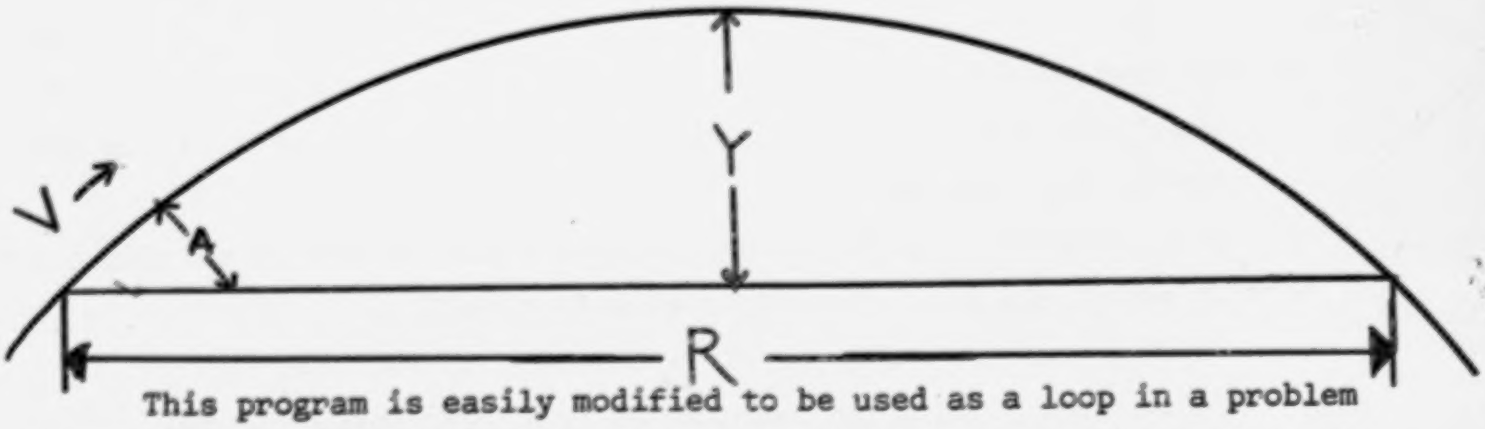
COMPUTER-ORIENTED SOLUTION OF A BALLISTIC PROBLEM

by Bill Tipton and Marvin Miller

The solution of problems involving two dimensional motion (non-straight line motion), wherein numerous computations must be made under a variety of given conditions, is readily adaptable to a computer program. Application of the computer gives a flexibility for dealing with a great number of changing situations in a given type of problem. The following example illustrates a basic situation wherein a body is projected with a given initial velocity in some direction as determined by the angle measured from the horizontal. As in most problems in the elementary stage, only those factors which are readily determined by secondary school laboratory techniques are considered. Thus air-friction, coriolis effect, etc, are not considered in this example. If these are to be introduced at a later stage of investigation, the basic program would remain the same, making it necessary only to add the instructions in the program to account for any changes of motion due to these factors.

This example will require the student to develop an expression such that the maximum range is a function of the initial velocity (V) and the angle of projection (A) if he wishes to work with only one function. In this example we wish to examine other variables, i.e., time of flight (T) and maximum altitude (Y); therefore, other functions are necessary. The conversion of radians to degrees (and having this result in tabular form) is also an extraneous portion of the program so far as determining the range; however, it may be of interest to illustrate the value of the angle in each of the two units.

As may be noted, the use of constants having seven place accuracy results in an error of four parts in ten thousand after ninety solution cycles. However, this result may be beneficial; as it illustrates the error introduced into a computation by use of rounded numbers.



This program is easily modified to be used as a loop in a problem involving other types of motion at the earth's surface. Also, the reader can either extend or limit the scope of the program as given here by making desired changes in the initial velocity or by reading in only those angle values for which he desires a solution. For example, if the initial velocity is 3000 cm/sec at an angle of 30 degrees, this would necessitate a change in the program as follows: Assign the new value (3000) in line number 10. Also assign the new value (30) in line number 50 and remove lines numbered 130 and 140. The resulting program would then determine the range, altitude, and the time of flight of the projectile for that particular angle of elevation and initial velocity.

The student should familiarize himself with the meaning of these formulas: $A = D(.01745329)$, where A is the measure of an angle in radians, and D is the measure of that angle in degrees. (The conversion factor used here is only approximate, not exact.)

$R = 2V^2 \cos A \sin A/G$, where V is the initial velocity (expressed in cm/sec), A is the measure of the launch angle (in radians), G is the gravitational force toward the earth (980 cm/sec^2), and R is the horizontal range of the projectle (in cm);

$T = R/V \cos A$, where T is the total time in flight (in seconds);

$Y = V \sin A\ (t) - \frac{1}{2} Gt^2$, where t is one-half of T (this would be the time elapsed from launch until the projectile reaches its apogee); and Y is the height of the apogee (the maximum height attained, in cm).

In the following discussion of the program, it has been assumed that the student is familiar with the BASIC language which is a very widely used medium for teletype communication with a time-sharing computer such as the G-E 235.

Having understood these formulas (and perhaps performed a few trial computations by conventional methods), the student must concern himself not only with (1)the problem of instructing the machine to substitute his data (known values for variables V, D, and G) into the above formulas, and subsequently calculate corresponding values for A, R, T, and Y, but also he will (2)want to program the necessary steps which will cause the machine to arrange the output information according to an orderly, easy-to-read format. And, as is mentioned elsewhere, if the student desires to test varying conditions in the launch angle and initial velocity, he should (3)create a loop in the program which will automatically cause the machine to repeat its cycle of computations until all the given data is exhausted.

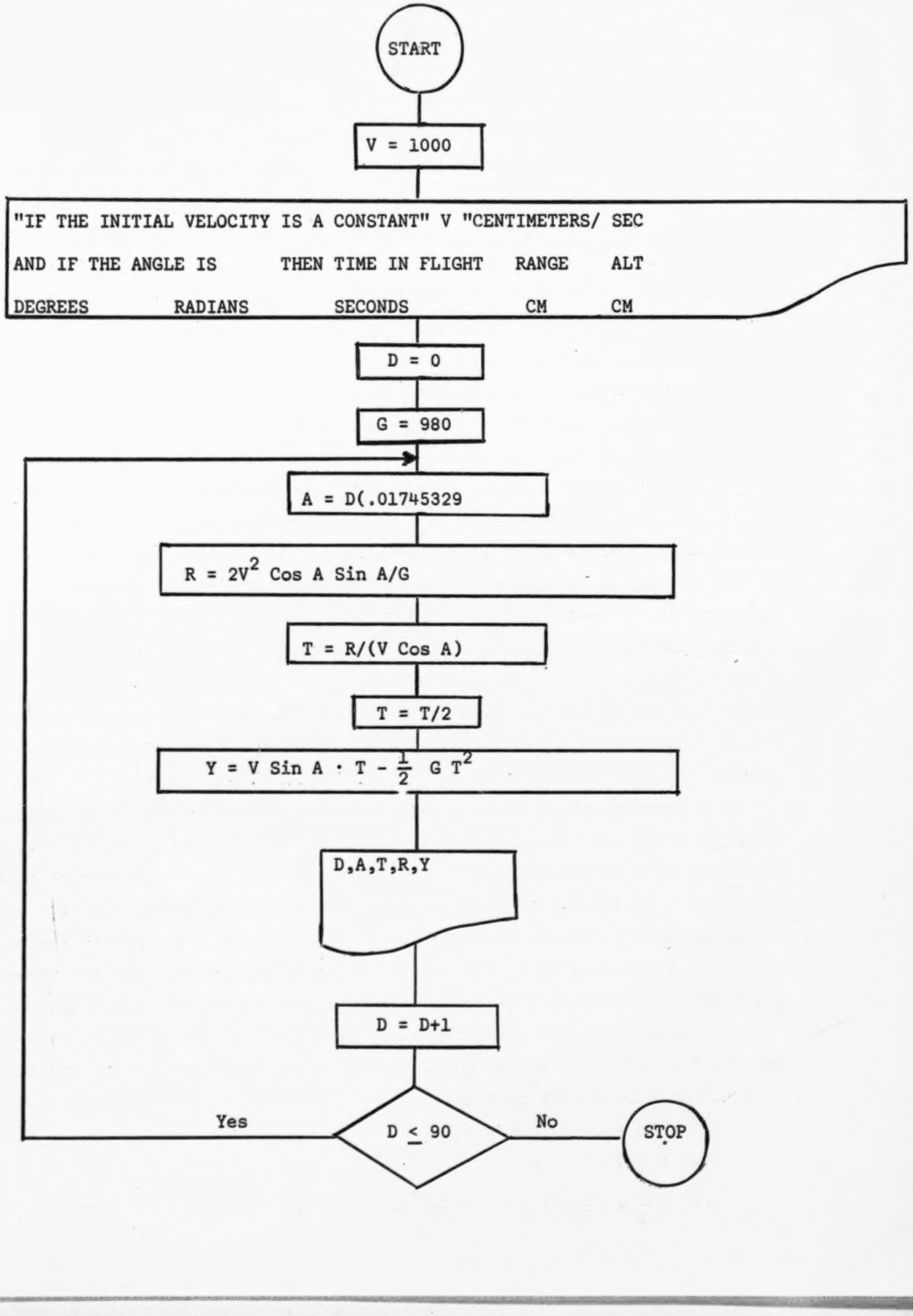
The part of the problem identified as (1) above (i.e., substituting known values for some variables and computing corresponding values for the other variables) is achieved by lines 10 and 50-110 in the program.

The instructions which cause the output to be printed in tabular form (objective 2 above) are lines 20-45 and 120. (Line 45 simply causes the teletype to skip one vertical line space).

Note that the commas in line 120 instruct the machine to space the columns of numerical output data in 15-space horizontal intervals. That is, the left sides of the 5 columns will begin at the first, six tenth, thirty-first, forty-sixth, and sixty-first spaces on the 75 character width of the printing space. Therefore, care must be taken when typing lines 30 and 40 to cause the column headings to be properly placed.

Objective 3 is fulfilled by lines 130 and 140. This loop is conditional - the computer continues to increase the measure of the launch angle by 1 degree until the angle becomes 91 degrees. The data resulting from such a case would duplicate that of an 89 degree angle. Therefore, we have chosen the condition that the angle exceed 90 degrees as that which directs the computer to "jump" out of the cycle of: (1) substituting & evaluating, (2) printing information, and (3) repeating with a larger angle measure, by line 150, the end statement.

```
10 LET V= 1000
20 PRINT"IF THE INITIAL VELØCITY IS A CØNSTANT"  V"CENTIMETERS/SEC"
30 PRINT" AND IF THE ANGLE IS    THEN TIME IN FLIGHT   RANGE      ALT"
40 PRING"DEGREES         RADIANS       SECØNDS          CM         CM"
45 PRINT
50 LET D=0
60 LET G=980
70 LET A = D*.01745329
80 LET R= 2*V↑2*CØS(A)*SIN(A)/G
90 LET T = R/(V*CØS(A))
100 LET T=T/2
110 LET Y=V*SIN(A)*T-1/2*G*T*T
120 PRINT D,A,T,R,Y
130 LET D=D+1
140 IF D<=90 THEN 70
150 END
RUN

BALIST     10:25   PX MØN 08/21/67

IF THE INITIAL VELØCITY IS A CØNSTANT 1000    CENTIMETERS/SEC
AND IF THE ANGLE IS      THEN TIME IN FLIGHT   RANGE      ALT
DEGREES        RADIANS        SECØNDS            CM         CM

 0              0              0              0              0
 1              1.74533 E-2    1.78086 E-2    35.6117        .155401
 2              3.49066 E-2    3.56117 E-2    71.1801        .621416
 3              5.23599 E-2    .053404        106.662        1.39748
 4              6.98132 E-2    7.11801 E-2    142.013        2.48263
 5              8.72665 E-2    8.89344 E-2    177.192        3.87557
 6              (etc.)
```


START
V = 1000
"IF THE INITIAL VELOCITY IS A CONSTANT" V "CENTIMETERS/ SEC
AND IF THE ANGLE IS THEN TIME IN FLIGHT RANGE ALT
DEGREES RADIANS SECONDS CM CM
D = 0
G = 980
A = D(.01745329
R = 2V2 Cos A Sin A/G
T = R/(V Cos A)
T = T/2
Y = V Sin A · T - 1/2 G T2
D,A,T,R,Y
D = D+1
Yes
D ≤ 90
No
STOP

A COMPUTERIZED APPROACH TO GRAPHING PARABOLAS

by Lavere C. Wilson

In this unit, functions of the form $[(x,y): ax^2 + bx + c = y]$ where *a*, *b*, and *c* are real numbers and $a \neq 0$ will be discussed. The graph of each such function is called a parabola. Although there are parabolas which are not graphs of functions, this discussion will involve only those which are.

The parabolas under discussion here have either a maximum or a minimum value, a vertex point, and an axis of symmetry. The parabola in Figure 1 has a minimum value of N, the coordinates of the vertex point are (M,N), and the axis of symmetry is defined by the equation $x = M$, where M is the x coordinate of the vertex point. In Figure 2, the vertex point and the axis of symmetry are the same as in Figure 1. The parabola in Figure 2 has a maximum value represented by N.

In the equation, $ax^2 + bx + c = y$, where $a \neq 0$, the coefficients *a*, *b*, and *c* determine certain things about the parabola. If $b = 0$ and $c = 0$, the resulting equation, $ax^2 = y$, is the equation of a parabola which has its vertex point at the origin, (0,0). The axis of symmetry is the y-axis, as in Figure 3. If $b = 0$, resulting in the equation $ax^2 + c = y$, the value *c* is positive, the curve is moved up *c* units; and if *c* is negative, the curve is moved down $|c|$ units. See Figures 4 and 5.

If $b \neq 0$ in the equation $ax^2 + bx + c = y$, then this causes a horizontal displacement of the parabola from a position symmetric with the y-axis.

In summary, the following may be said about the effect of the values of *a*, *b*, and *c* upon the graph of the function $[(x,y): ax^2 + bx + c = y, a = 0]$. The value of *a* determines whether the parabola will have a maximum or minimum value. If $a < 0$, the parabola will have a maximum value, and if $a > 0$ the parabola will have a minimum value. The value of *b* determines whether the axis of symmetry is to the left or right of the y-axis, and the value of *c* affects the vertical position of the parabola, moving its graph up or down.

By completing the square with the equation $ax^2 + bx + c = y$, $(a \neq 0)$, information about the vertex point, the maximum or minimum value, and the axis of symmetry may be obtained.

$$ax^2 + bx + c = y \text{ where } a \neq 0 \qquad (1)$$

$$a(x^2 + \frac{b}{a}x) + c = y \qquad (2)$$

$$a(x^2 + \frac{b}{a} + (\frac{b}{2a})^2) + c - (\frac{b}{2a})^2 = y \qquad (3)$$

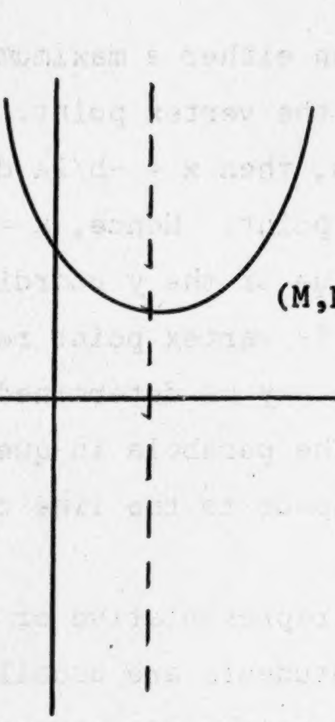


Figure 1

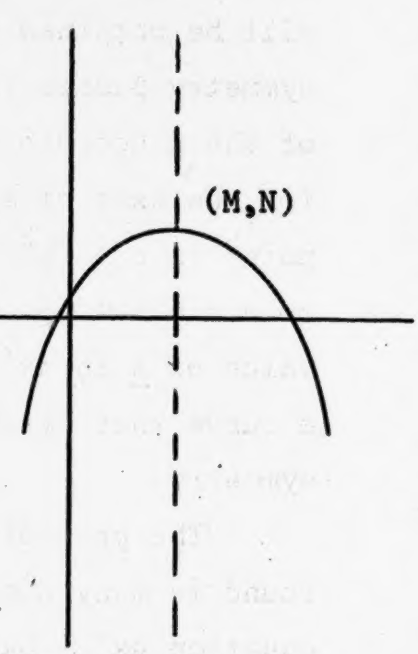


Figure 2

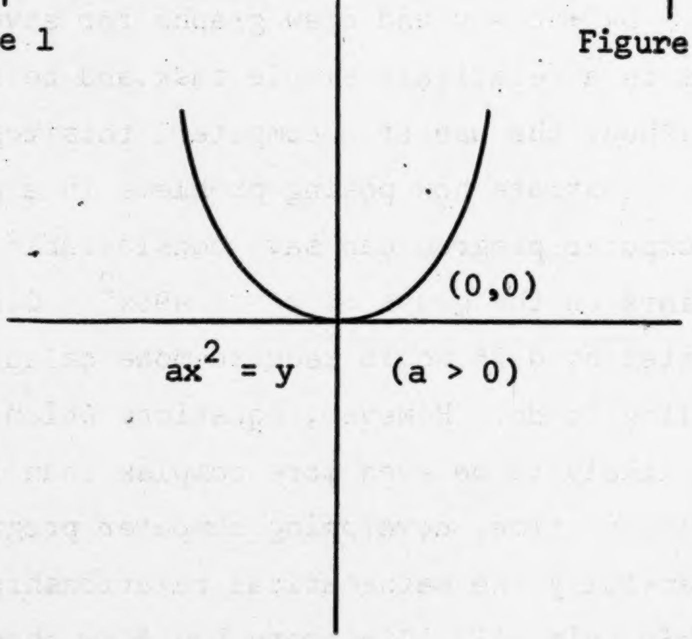


Figure 3

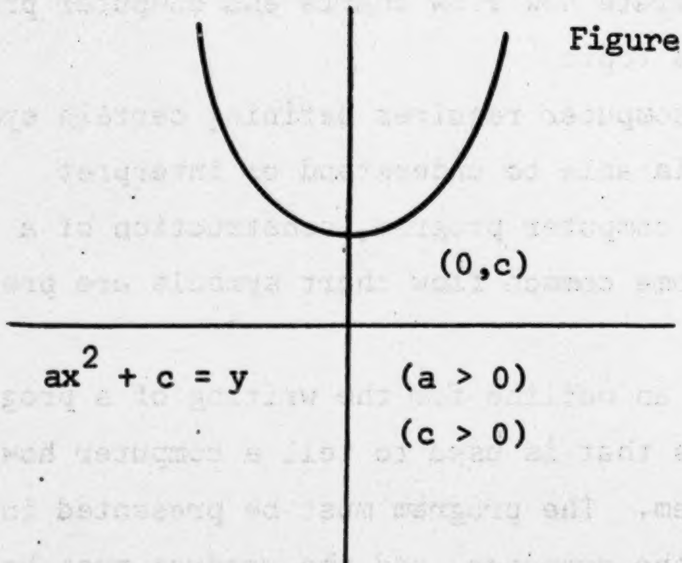

(0,c)
$ax^2 + c = y$
$(a > 0)$
$(c > 0)$

Figure 4

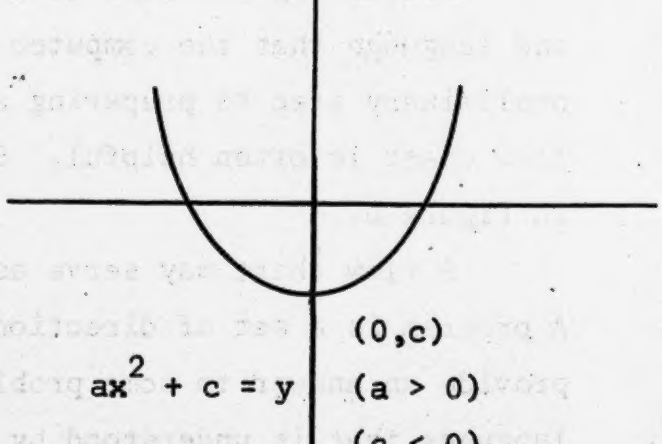

(0,c)
$ax^2 + c = y$
$(a > 0)$
$(c < 0)$

Figure 5

$$a(x + \frac{b}{2a})^2 + c - (b^2/4a^2) = y \qquad (4)$$

In equation (4), if $x = -b/2a$, then either a maximum or a minimum value will be obtained which determines the vertex point. Since the axis of symmetry passes through the vertex, then $x = -b/2a$ determines the value of the x coordinate of the vertex point. Hence, $x = -b/2a$ is the equation for the axis of symmetry. The value of the y coordinate of the vertex point is $c - (b^2/4a^2)$. Whether this vertex point represents a maximum or a minimum point of the parabola may be determined by observing the value of $\underline{a}$ in $ax^2 + bx + c = y$. The parabola in question will always be a curve that is symmetric with respect to the line called the axis of symmetry.

The preceding discussion is representative of the type of discussion found in many Algebra II texts. Students are usually asked to analyze the equation $ax^2 + bx + c = y$ and draw graphs for several specific examples. Although this is a relatively simple task and certainly can be accomplished quite well without the use of a computer, this topic does provide an opportunity to illustrate how posing problems in a general form and developing a computer program can save considerable calculation time. To calculate points on the graph of $y = 1.495x^2 - 0.832x + 15.632$ for values of x incremented by 0.25 would require more calculation than most students would be willing to do. However, equations which arise from "real" problems are likely to be even more complex than this. In addition to saving calculation time, developing computer programs encourages students to analyze carefully the mathematical relationships involved. The remainder of this unit will illustrate how flow charts and computer programs could be used in presenting this topic.

Presenting problems to a computer requires defining certain symbols and language that the computer is able to understand or interpret. As a preliminary step to preparing a computer program, construction of a flow chart is often helpful. Some common flow chart symbols are presented in Figure 6.

A flow chart may serve as an outline for the writing of a program. A program is a set of directions that is used to tell a computer how to provide an answer to some problem. The program must be presented in a language that is understood by the computer, and the program must be completely and precisely stated. Some of the programming symbols are listed in Figure 7. A program designed to yield information for plotting the graph of the equation $ax^2 = y$ could be written as indicated by Program I.

FLOW CHART SYMBOLS

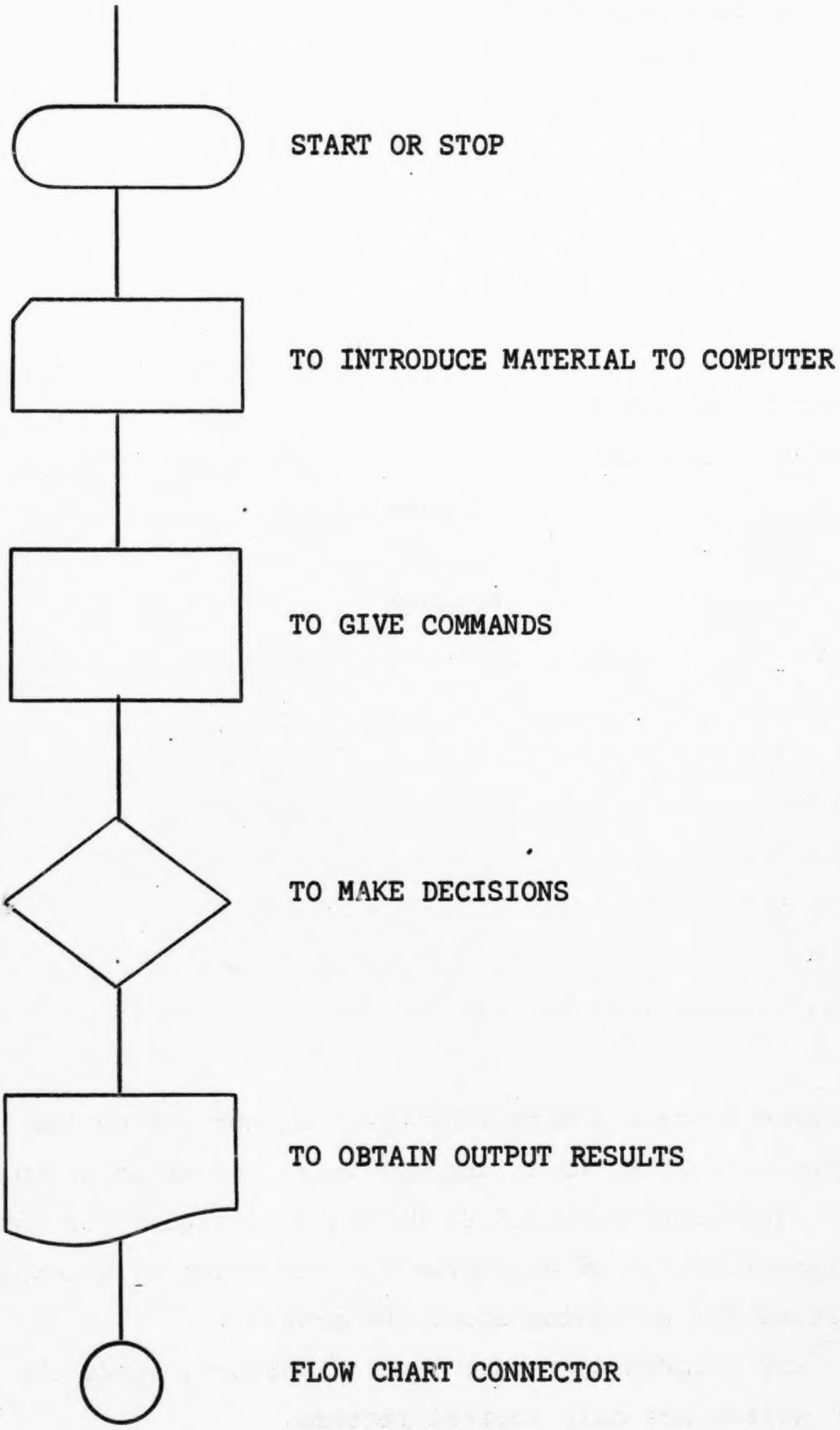

START OR STOP
TO INTRODUCE MATERIAL TO COMPUTER
TO GIVE COMMANDS
TO MAKE DECISIONS
TO OBTAIN OUTPUT RESULTS
FLOW CHART CONNECTOR

Figure 6

PROGRAMMING SYMBOLS

=	IS EQUAL TO
<	IS LESS THAN
<=	IS LESS THAN OR EQUAL TO
>	IS GREATER THAN
>=	IS GREATER THAN OR EQUAL TO
<>	IS NOT EQUAL TO
+	ADDITION
-	SUBTRACTION
*	MULTIPLICATION
/	DIVISION
↑	RAISE TO THE POWER
( )	FORM OF INCLUSION

Figure 7

PROGRAM I

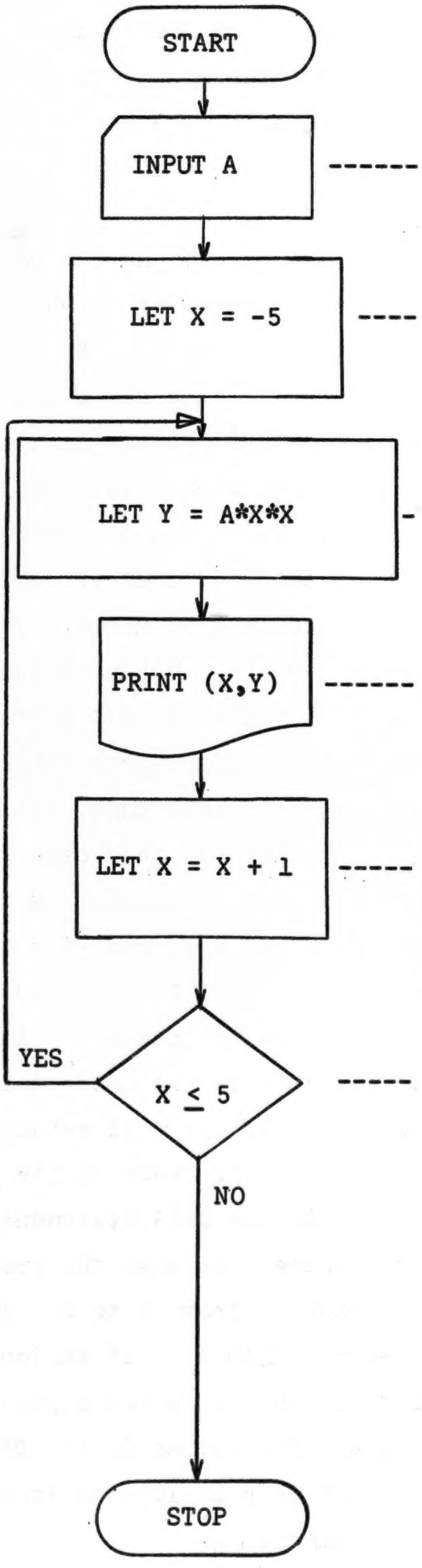
```
10 READ A
20 LET X = -5
30 LET Y = A*X*X
40 PRINT X,Y
50 LET X = X + 1
60 IF X <= 5 THEN 30
70 GØ TØ 10
80 DATA 1,-1,2,-2,3,-3,.5,-.5,.25,-.25
90 END
```

Compare Program I with Flow Chart I, and notice the similarities. The program is written in a language which resembles ordinary mathematical notation. This language, called BASIC, is designed for complete and precise specification of a program for execution by a computer.

Observe the following about the program:

1. The program uses only capital letters, since the teletypewriter has only capital letters.
2. Each line of the program begins with a number. These numbers are called <u>line numbers</u> and serve to identify the lines, each of which is called a <u>statement</u>. Line numbers serve to specify the order in which the statements are to be performed by the computer.
3. Each statement starts, after its line number, with an English word. This word denotes the type of statement.

FLOW CHART I



THIS STEP INTRODUCES THE VALUE OF A.

THIS STEP ASSIGNS THE ORIGINAL VALUE OF X.

THIS STEP CALCULATES $Y = AX^2$.

THIS STEP CALLS FOR THE PRINTING OF (X,Y).

THIS STEP INCREASES THE ORIGINAL VALUE OF X BY ONE UNIT. (THIS IS NOT AN EQUATION. THE EQUALS SIGN IS USED HERE TO MEAN THE FOLLOWING: REPLACE THE PREVIOUS VALUE ASSIGNED TO THE VARIABLE X BY THAT VALUE PLUS ONE.)

THIS STEP CHECKS TO SEE IF $X \leq 5$ AFTER X HAS BEEN INCREASED.

a. IF YES, IT REFERS BACK TO THE CALCULATION PORTION OF THE CHART TO HAVE THE NEW VALUE OF Y CALCULATED. THIS PART OF THE CHART IS CALLED A LOOP. THE FIRST TIME THROUGH THE LOOP THE VALUE OF X IS -5, THE SECOND TIME THROUGH THE VALUE OF X IS -4, THIRD TIME, -3, FOURTH -2, AND SO ON UNTIL X = 5. THE VALUE OF Y FOR X = 5 IS CALCULATED AND THEN THE STEP "LET X = X + 1" SETS X = 6. SINCE 6 IS NOT LESS THAN OR EQUAL TO 5 THIS ENDS THE LOOP AND LEADS TO THE STOP STATEMENT.

b. IF NO, IT GIVES DIRECTIONS TO STOP.

A line-by-line analysis of the program will now be presented. The first statement, 10, is a READ statement. It must be accompanied by one or more DATA statements. When the computer encounters a READ statement while executing a program, it will cause the variables listed after the READ to be given values according to the next available numbers in the DATA statements. In this case, A is read in statement 10; and the first value in the line 80 DATA statement is assigned to A. That is, A is assigned the value 1.

The computer then proceeds to statement 20 and assigns the value -5 to the variable x. The computer then proceeds to statement 30 where it is directed to compute a value of y according to the formula $ax^2 = y$. (The asterisk is used to denote multiplication.) In general, a LET statement directs the computer to set a variable equal to the numerical value of the formula on the right side of the equal sign. The computer then proceeds to line 40 where it is directed to print x and y, which in this case are -5 and 25, respectively. The computer then proceeds to line 50 where it is directed to replace the value of x by x + 1. In this case, it replaces -5 by -4. Now x = -4. The computer then proceeds to line 60 where it encounters the statement, IF x <= 5 THEN 30. If the answer to the question "Is x <= 5?" is "No," the computer goes on to the next statement. If the answer is "Yes," as it is in this case, the computer goes to the line number stated after "THEN," in this case line 30. This begins a "loop" which will result in the computer calculating values of $ax^2 = y$ and printing them out for values of x ranging from -5 to 5. Now, assume the computer is at line 40 and prints out 5,25. It now proceeds to line 50 and assigns x the value 6. It then proceeds to line 60. Since 6 is not less than or equal to 5, it proceeds to the next line in the program. Line 70 directs the computer to "GØ TØ 10." It returns to line 10, where it is directed to "READ A." It then proceeds to the first DATA statement and reads the next value listed in the DATA statement, which is in this case -1. Now the computer proceeds through the program again printing out values of $-1x^2 = y$ for x ranging from -5 to 5. It then returns to line 10 where it is directed to "READ A." It assigns A the next unused value in the DATA statement, in this case 2, and proceeds through the program again. After assigning all the values in the DATA statement to A, the computer will print "OUT OF DATA IN 10" and stop. Every program must terminate with an END statement.

This program could be used by students to produce sets of ordered pairs for graphing. Students could draw the graphs and discuss the effect of changes in A on the graph of $ax^2 = y$. Time saved in computation could be used for discussion of flow charting, programming, and analysis of graphs.

PROGRAM II

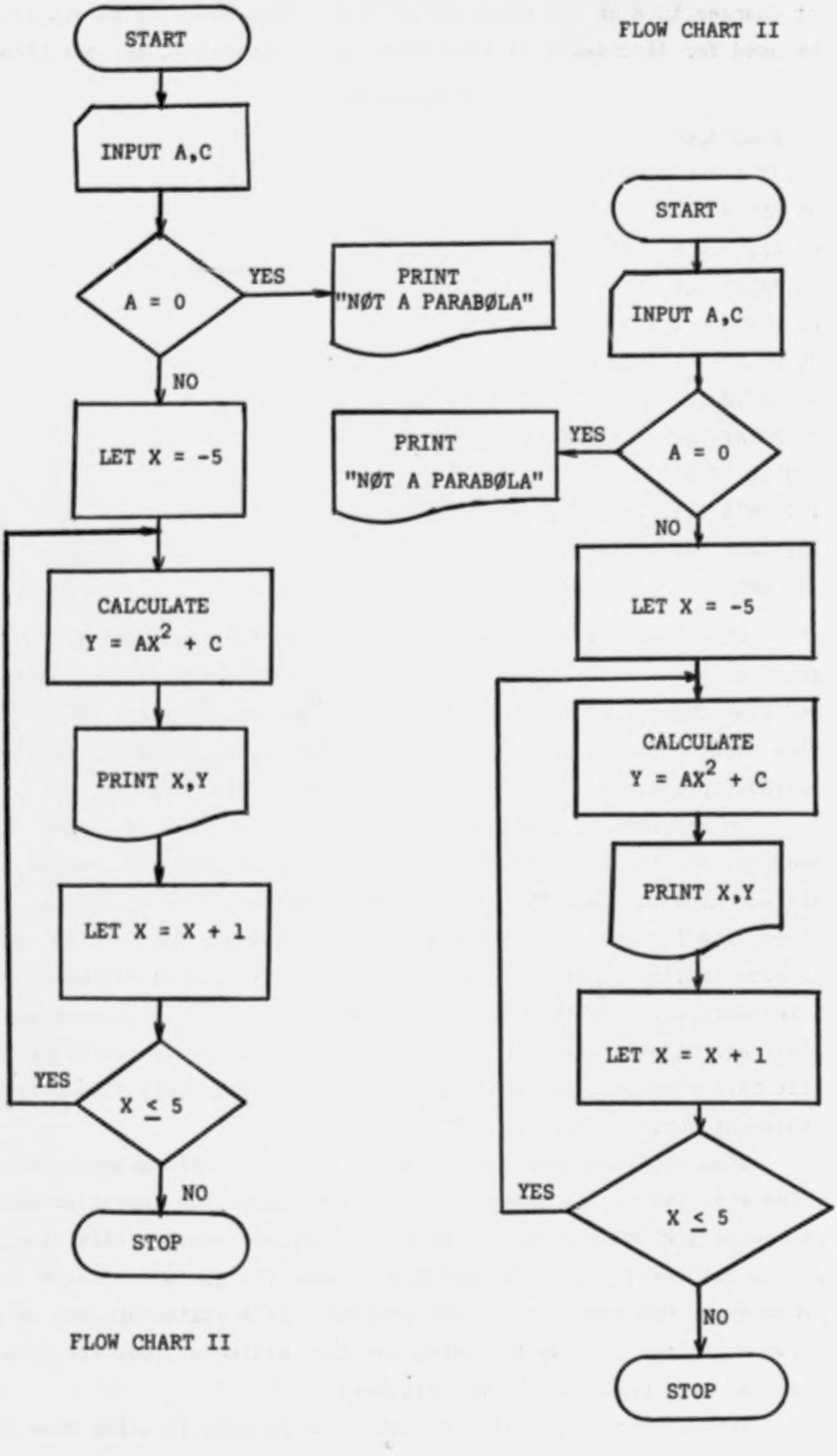
```
10 READ A,C
20 IF A = 0 THEN 90
30 LET X = -5
40 LET Y = A * X↑2 + C
50 PRINT X,Y
60 LET X - X + 1
70 IF X <= 5 THEN 40
80 GØ TØ 10
90 PRINT "NOT A PARABOLA"
100 GØ TØ 10
110 DATA 1,5,1,-5,2,7,2,-7,.5,7.25,-.5,-7.25
120 DATA -2,-10,-4.5,20
130 END
```

Flow Chart II and Program II are for calculating points on the graph of $ax^2 + c = y$. The only real difference between Flow Chart II and Flow Chart I is the inclusion of a check on the value of A. This flow chart could be presented in blank form to students on a ditto or overhead projector and completed as a class project.

In Program II, when the computer starts on line 10, "READ A,C," it searches for the first DATA statement in the program and assigns the first listed value to A and the next listed value to C. In this case, it would go to line 110 and pick up the value 1 for A and 5 for C. The next time it gets to line 10 it would assign the next two listed values to A and C respectively. In this case A = 1, and C = -5. This process continues until all DATA in line 110 is used. Then the computer continues to the next DATA statement and assigns values to A and C respectively from that statement until it runs out of data.

Line 40 introduces the arrow "↑" which, in this example, means raise x to the second power. Line 90 introduces the quotation marks following a PRINT statement. When the computer reaches this line, it prints out exactly what is enclosed between the quotation marks and proceeds to the next line in the program. DATA statements may be placed anywhere in the program following the READ statement, but are usually inserted just prior to the END statement.

Students could be asked to construct Program II using Flow Chart II as a guide. Graphs obtained from the selected data could be analyzed to


START
INPUT A,C
A = 0
YES
PRINT "NØT A PARABØLA"
NO
LET X = -5
CALCULATE Y = AX^2 + C
PRINT X,Y
LET X = X + 1
X ≤ 5
YES
NO
STOP
START
INPUT A,C
A = 0
YES
PRINT "NØT A PARABØLA"
NO
LET X = -5
CALCULATE Y = AX^2 + C
PRINT X,Y
LET X = X + 1
X ≤ 5
YES
NO
STOP

FLOW CHART II

FLOW CHART II

determine the effect of changes in the values of A and C.

PROGRAM III

Dotted lines indicate missing parts of this program. Complete the program using Flow Chart III as a guide.

```
10 READ A,B,_ _ _
20 IF _ _ _ _ _ THEN 90
30 LET X = -B/(2 * _ _ _ ) - 5
40 LET Y = A * X * X + _ _ _ _ _ + C
50 PRINT _ _ _ _ _
60 LET _ _ _ _ _ _ _ _ _ _ _
70 IF X <= -B/(2 * A) + 5 THEN _ _ _
80 GØ TØ 10
90 PRINT "_ _ _ _ _ _ _ _ _ _ _"
100 GØ TØ 10
110 DATA 1,-4,4,1,4,4,1,-8,16,1,8,16
120 DATA 2,-7,0,2,7,0,-2,-7,7,-2,7,15
130 DATA 3.9,-4.2,-27.6
140 _ _ _ _
```

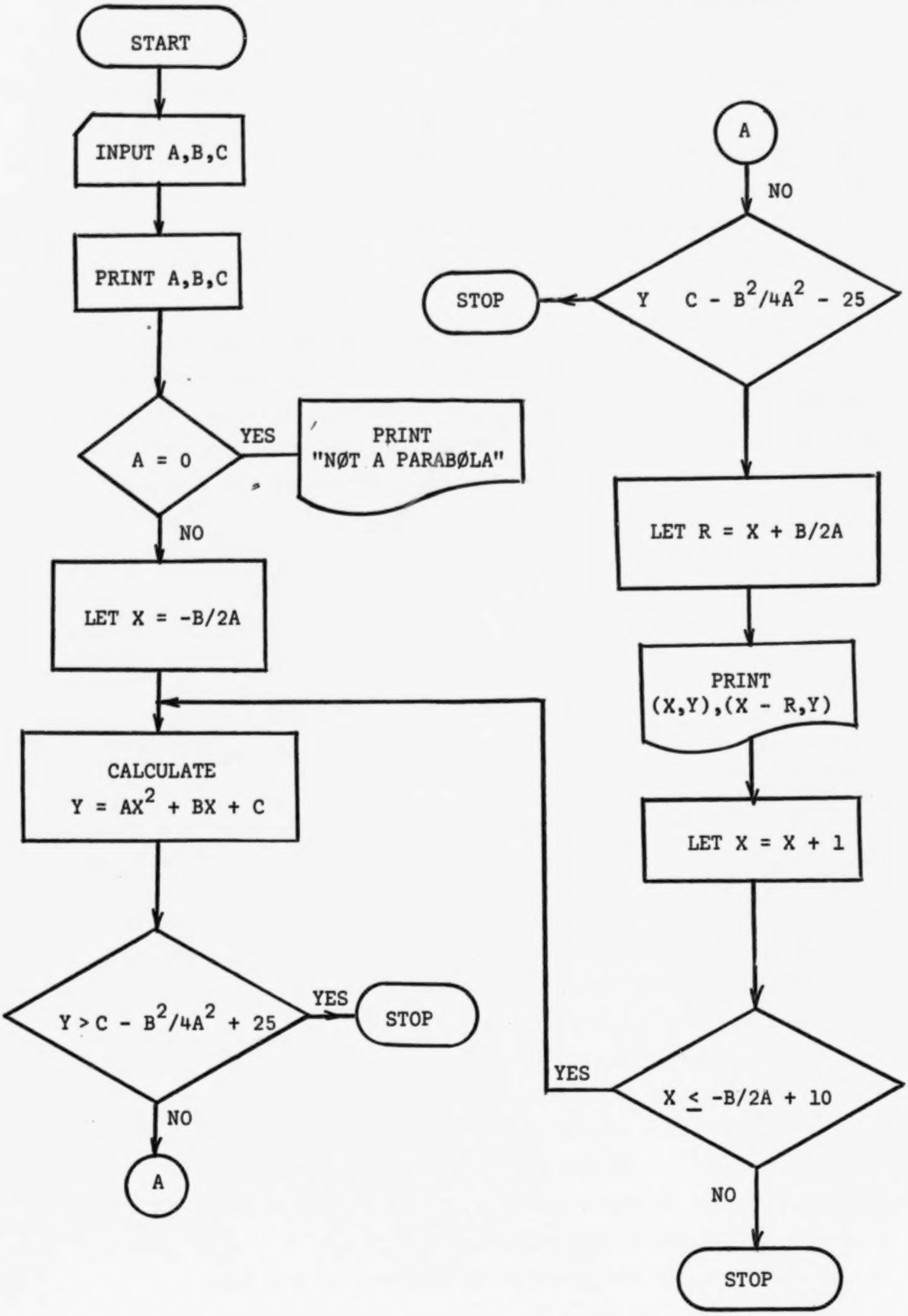
PROGRAM IV

```
10 READ A,B,C
15 PRINT A,B,C
20 IF A = 0 THEN 130
30 LET X = -B/(2 * A)
40 LET Y = A * X * X + B * X + C
50 IF Y > C - B * B/(4 * A * A) + 25 THEN 10
60 IF Y < C - B * B/(4 * A * A) - 25 THEN 10
70 LET R = X + B/(2 * A)
80 PRINT "("X","Y")"
90 PRINT "("X - R","Y")"
100 LET X = X + 1
110 IF X <= -B/(2*A) + 10 THEN 40
120 GØ TØ 10
130 PRINT "NØT A PARABØLA"
140 GØ TØ 10
150 DATA 1,4,4,0,2,3,1,0,0,-3,2,-5,-3.2,-8,14
160 END
```

The axis of symmetry of a parabola whose equation is $ax^2 + bx + c = y$ is determined by the equation $x = -b/2a$. Flow Chart III and Computer Program III provide for computation of ordered pairs with integral values of x which lie in the region five units on either side of the axis of symmetry of the parabola whose equation is $ax^2 + bx + c = y$. Programs I and II calculated values of y for the interval -5 <= x <= 5. Program III calculates values of y for the interval (-b/2a) - 5 = x = (-b/2a) + 5. Flow Chart IV and Program IV specify limits on the value of y, as well as x. Any limitation could be chosen. In Program IV y is calculated for integral values of x <= -b/2a + 10 so long as the value of y is less than or equal to the parabola's minimum value plus 25 or greater than or equal to the parabola's maximum value minus 25. Symmetry of the curve is used to produce pairs of points rather than one point at a time.

In order to be very proficient in programming and working with computers, it is essential to have a knowledge of mathematics. A computer can perform many operations very rapidly, but it can only do what it is told to do. Analyzing a problem carefully in terms of its mathematical structure and then writing a program to perform the necessary calculations can save a great deal of tedious computation and also leads to a more thorough understanding of the mathematics involved.

FLOW CHART IV


START
INPUT A,B,C
PRINT A,B,C
A = 0
YES
PRINT "NØT A PARABØLA"
NO
LET X = -B/2A
CALCULATE Y = AX^2 + BX + C
Y > C - B^2/4A^2 + 25
YES
STOP
NO
A
A
NO
Y   C - B^2/4A^2 - 25
STOP
LET R = X + B/2A
PRINT (X,Y),(X - R,Y)
LET X = X + 1
YES
X ≤ -B/2A + 10
NO
STOP

APPLYING THE COMPUTER TO A UNIT ON CIRCLES IN GEOMETRY

by Karen L. Winquist

This project is an application of the computer to a unit on circles in geometry. The problems and programs presented here are meant to be supplementary materials which will enrich and expand a unit on circles. While also stimulating the student's interest in the subject matter, it is hoped the material presented will motivate the students to make up programs of their own and to investigate many other problems connected with circles and also spheres.

Prior to presenting the problems here, the students should have studied the definitions and proofs of the theorems and corollaries involved in a unit on circles. It is furthermore assumed that they have a knowledge of BASIC language and are familiar with programming elementary problems and working on the time sharing system of the computer.

Definitions that Should Be Known for a Unit on Circles

1. Circle
2. Sphere
3. Center of circle
4. Radius of circle
5. Diameter of circle
6. Concentric circles
7. Chord
8. Secant
9. Interior of circle
10. Exterior of circle
11. Tangent to a circle
12. Tangent plane to a sphere
13. Tangent circles -- externally and internally
14. Congruent circles
15. Central angle of a circle
16. Minor and major arcs
17. Semicircle
18. Angle inscribed in an arc
19. Angle intercepting an arc
20. Quadrilateral inscribed in a circle
21. Quadrilateral circumscribed about a circle

Special Theorems that Should Be Known for Working with the Problems of this Project

1. The Power Theorem -- Given a circle C, and a point Q of its exterior -- let L_1 be a secant line through Q, intersecting C in points R and S. Let L_2 be another secant line through Q, intersecting C in points U and T. Then (QR)(QS) = (QU)(QT).
2. Given a tangent segment $\overline{QT}$ to a circle, and a secant line through Q, intersecting the circle in points R and S -- then $(AR)(QS) = QT^2$.
3. Let $\overline{RS}$ and $\overline{TU}$ be chords of the same circle, intersecting at Q -- then (AR)(QS) = (QU)(QT).
4. The graph of the equation $(x - a)^2 + (y - b)^2 = r^2$ is the circle with center (a, b) and radius r.
5. Every circle is the graph of an equation of the form $x^2 + y^2 + Ax = By + C = 0$.
6. The graph of the equation $x^2 + y^2 + Ax + By + C = 0$ is (1) a circle, (2) a point, or (3) the empty set.

Program 1 -- Finding the Coordinates of Points of a Circle with its Center at the Origin

The graph of the equation $X^2 + Y^2 = R^2$ is a circle with center at the origin and radius R. The following program will output coordinates of points belonging to the circle for X = 0 up to the length of the radius in steps of N. Lines 80 and 90 of the program can be changed for various radii and steps. The output is in the first quadrant and will be printed in two vertical columns labeled X and Y. It is not necessary to have the computer print the values for the other three quadrants as they can be figured as follows:

-X,Y for the second quadrant
-X,-Y for the third quadrant
X,-Y for the fourth quadrant

One example of a result is as follows for a circle of radius 3 figured where step N = .2.

```
X                      Y

0                      3
 .2                    2.99333
 .4                    2.97321
 .6                    2.93939
 .8                    2.89137
1.                     2.82843
1.2                    2.74955
1.4                    2.6533
1.6                    2.53772
1.8                    2.4
2.                     2.23607
2.2                    2.03961
2.4                    1.8
2.6                    1.49666
2.8                    1.07703
3                      2.72958E-4
```

Notice that in line 40 when X = 3 with R = 3, Y should = 0; however, the computer outputs the square root of 0 - 2.72958E-4, a very small number close to zero. This can be eliminated by inserting the following:

```
45 LET Z = INT(1000*Y)/1000
50 PRINT X,Z
```

Now all numbers will have three decimal places, giving Y = 0 for X = 3.

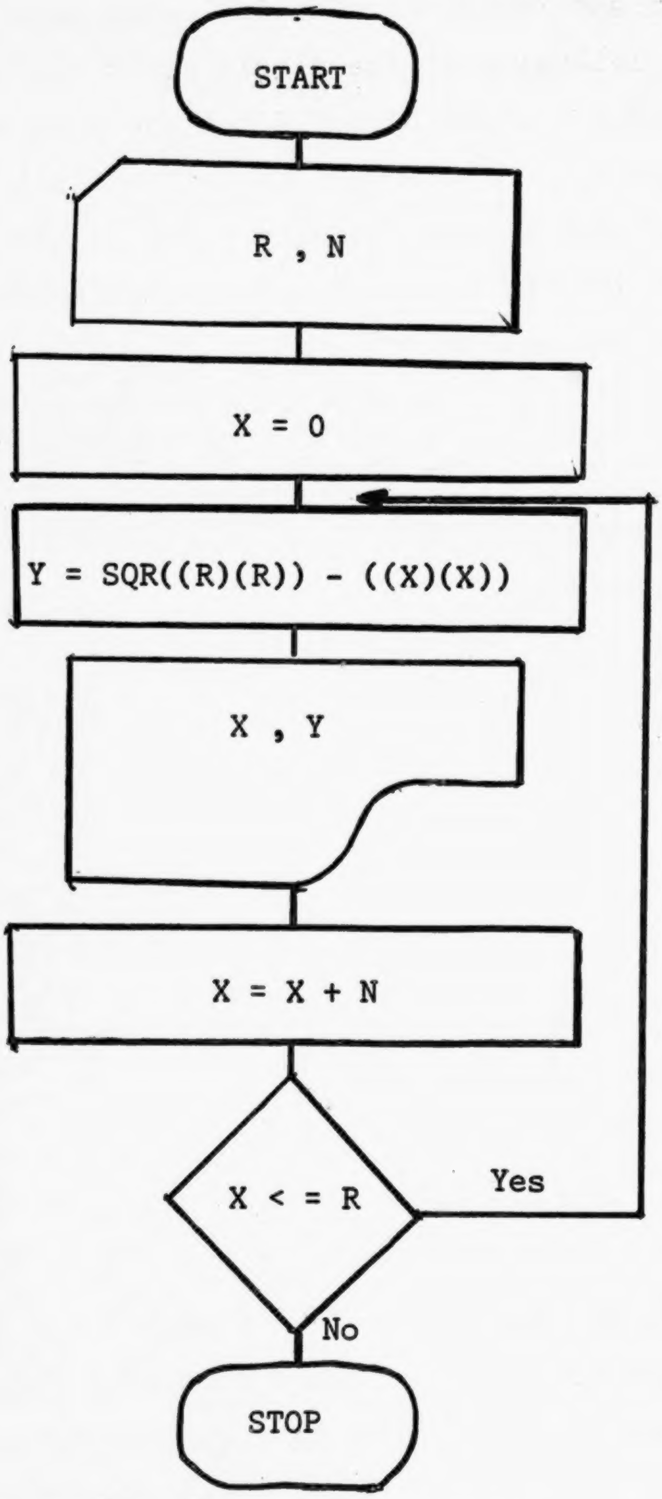
```
 10 READ R,N
 20 PRINT "X","Y"
 30 FØR X = 0 TØ R STEP N
 40 LET Y = SQR(R↑ 2-(X*X))
 50 PRINT X,Y
 60 NEXT X
 70 GØ TØ 10
 80 DATA
 90 DATA
100 END
```

Optional for 3 decimal place numbers:

```
45 LET Z = INT(1000*Y)/1000
50 PRINT X,Z
```

FLOW CHART #1

Flow Chart 1



Program 2 -- The Graph of $X^2 + Y^2 + AX + BY + C = 0$

As proved by a theorem, the equation of all circles is of the form $X^2 + Y^2 + AX + BY + C = 0$; however, conversely, $X^2 + Y^2 + AX + BY + C = 0$ is not always the equation of a circle. The following program shows that the converse is not always true. A, B, and C are the coefficients of X, Y, and the constant respectively. Inputing A, B, and C from any equation of the above standard form this program will print what kind of graph would be obtained from that equation? As one of the theorems stated, the graph will be one of three possibilities:

(1) a circle
(2) a point
(3) the empty set

if it is a circle, the computer will print the coordinates of its center and the length of its radius. If it is a point, the computer will print the coordinates of the point.

It is assumed here that in studying the proof of the above theorem, the student learned the various conditions yielding a circle, point, or the empty set from the expression $\frac{A^2 + B^2 - 4C}{4}$. However, if necessary as a review, the expression is derived as follows:

$$X^2 + Y^2 + AX + BY + C = 0$$

$$X^2 + Y^2 + AX + BY = -C$$

$$X^2 + AX + Y^2 + BY = -C$$

Completing the square: $[X^2 + AX + (\frac{A}{2})^2] + [Y^2 + BY + (\frac{B}{2})^2] = -C + (\frac{A}{2})^2 + (\frac{B}{2})^2$

$$(X + \frac{A}{2})^2 + (Y + \frac{B}{2})^2 = \frac{A^2}{4} + \frac{B^2}{4} - C$$

$$(X + \frac{A}{2})^2 + (Y + \frac{B}{2})^2 = \frac{A^2 + B^2 - 4C}{4}$$

As proved by a theorem, the above expression is the equation of a circle with its center at the point (-A/2,-B/2) with radius equal to the square root of $(\frac{A^2 + B^2 - 4C}{4})$.

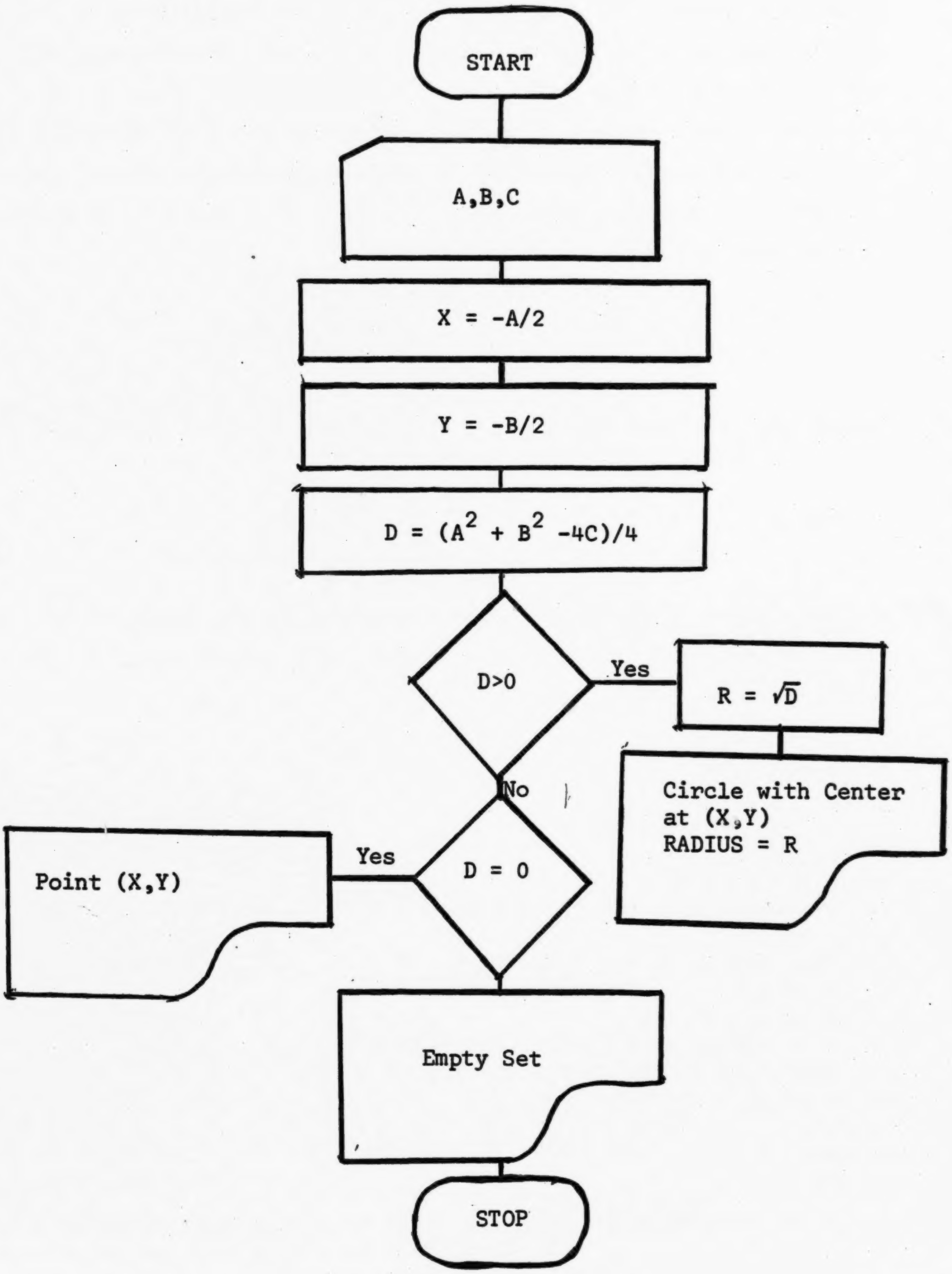
```
 10 READ A,B,C
 20 LET X= -A/2
 30 LET Y= -B/2
 40 LET D=(A*A+B*B-4*C)/4
 50 IF D>0 THEN 110
 60 IF D=0 THEN 90
 70 PRINT "GRAPH IS THE EMPTY SET"
 80 GØ TØ 10
 90 PRINT "GRAPH IS THE PØINT" X,Y
100 TØ TØ 10
110 LET R=SQR(D)
120 PRINT "GRAPH IS A CIRCLE WITH CENTER AT" X,Y
130 PRINT "THE RADIUS ØF THE CIRCLE IS" R
140 GØ TØ 10
150 DATA
160 DATA
170 END
```

Suggested data might be

-6,-8,21	Will be circle with center at (3,4) R = 2
8,-2,-8	Will be circle with center at (-4,1) R = 5
6,-2,10	Point (-3,1)
10,-4,33	Empty Set

FLOW CHART #

Flow Chart 2


START
A,B,C
X = -A/2
Y = -B/2
D = (A^2 + B^2 -4C)/4
D>0
Yes
R = √D
No
Circle with Center at (X,Y) RADIUS = R
D = 0
Yes
Point (X,Y)
Empty Set
STOP

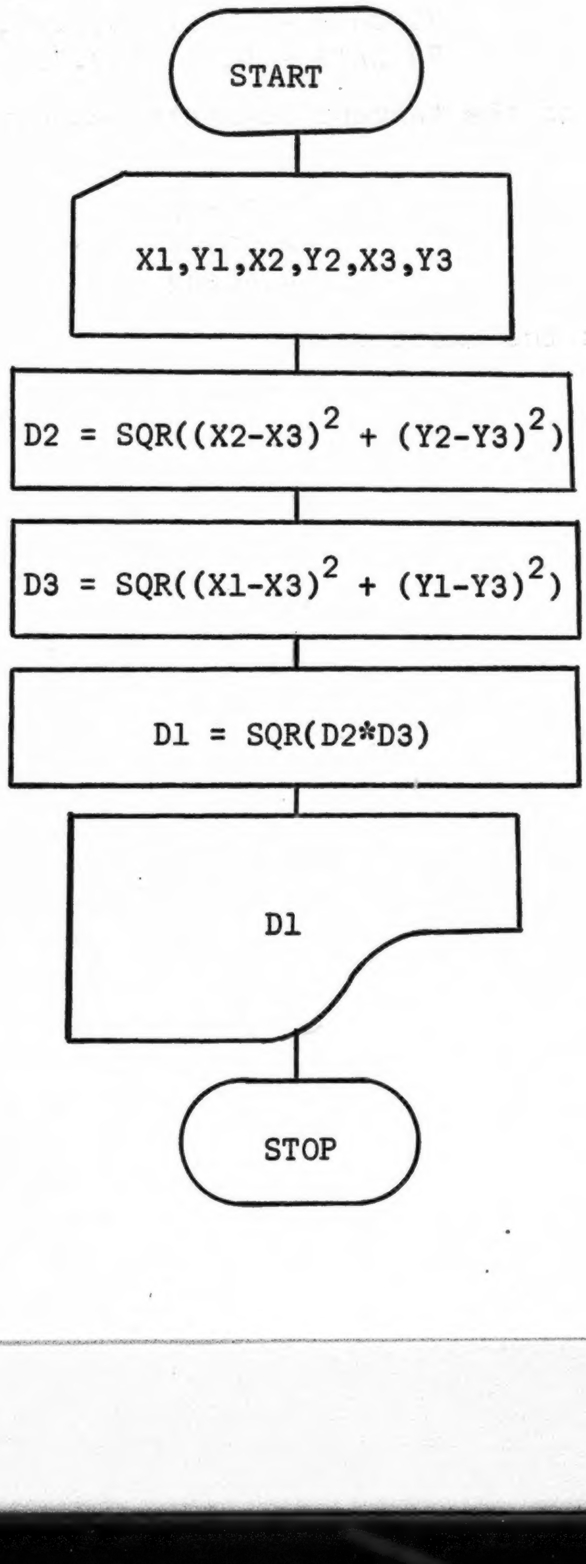
Program 3 -- Finding the Length of a Tangent Segment Using an Application of the Power Theorem

This program will give the length of a tangent segment $\overline{AB}$ to a circle knowing the coordinates of points D, C, and B of secant segment $\overline{DB}$. $\overline{DB}$ can be any secant segment to the circle from point B, intersecting the circle at points D and C as shown in Figure A. In the program (X1,Y1) must be the coordinates of point D, (X2,Y2) the coordinates of point C, and (X3,Y3) the coordinates of point B.

D2 = BC (distance from point B to point C)
D3 - BD (distance from point B to point D)
D1 = AB (distance from point A to point B or the length of the tangent)

If D2 and D3 distances are already known, then they can be substituted in lines 20 and 30.

```
10 READ X1,Y1,X2,Y2,X3,Y3
20 LET D2 = SQR((X2-X3)↑2+(Y2-Y3)↑2)
30 LET D3 = SQR((X1-X3)↑2+(Y1-Y3)↑2)
40 LET D1 = SQR(D2*D3)
50 PRINT "LENGTH ØF THE TANGENT SEGMENT = "D1
60 GØ TØ 10
70 DATA
80 DATA
90 END
```


START
X1,Y1,X2,Y2,X3,Y3
D2 = SQR((X2-X3)^2 + (Y2-Y3)^2)
D3 = SQR((X1-X3)^2 + (Y1-Y3)^2)
D1 = SQR(D2*D3)
D1
STOP

Flow Chart 3

Program 4 -- Showing the Tangent Length to Remain Constant for a Given Circle and Given External Point

In this problem, use the same program as that used in Program 3. The computer will show that the length of the tangent segment remains the same for various secants drawn to the circle from the given external point.

As shown in the diagram, Figure B, keep point B constant; and find the coordinates of D and C for various rotations of the secant. Program 1 may be helpful in getting the coordinates more exact. Using Program 3, place these coordinates in the data statements 70-80. The output will show the tangent length remaining fairly constant. It is very difficult to find the exact coordinates for all rotations of the secant; and, therefore, the tangent lengths will not be equal but will be fairly close, showing the student that secant rotations from point B do not affect the tangent length. One example that could be used as shown on the diagram is as follows (the constant point B being (5,0)):

```
70 DATA -3,0,3,0,5,0
71 DATA -.5,2.95,2.75,1.25,5,0
72 DATA -2.75,1.25,2.95,.33,5.0
73 DATA -.5,-2.95,2.75,-1.25,5,0
```

Lengths of the tangent segments respectively, then, are:

```
4
4.00802
4.03733
4.00802
```

The 4 is the exact value.

Figure A

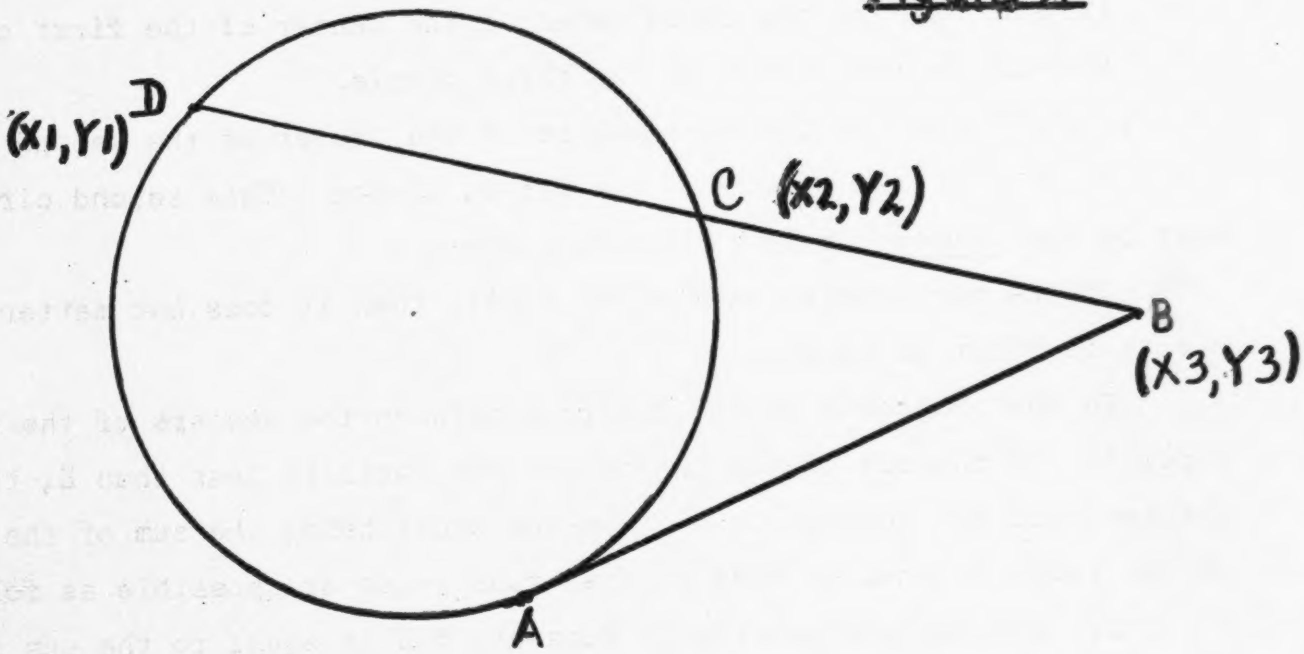
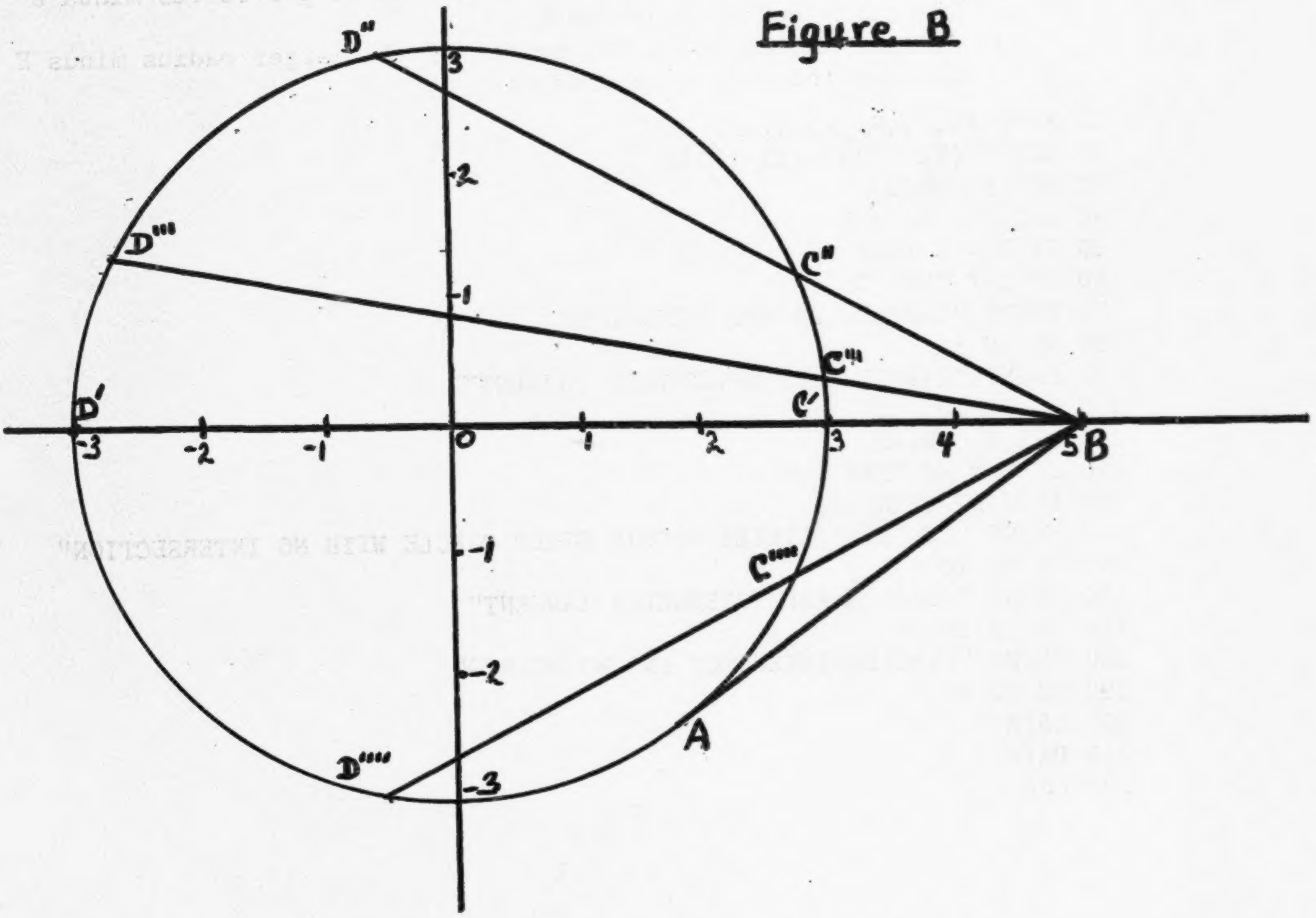

D
(X1,Y1)
C
(X2,Y2)
B
(X3,Y3)
A

Figure B


D''
3
2
D'''
C''
1
C'''
D'
C'
-3
-2
-1
0
1
2
3
4
5
B
-1
C''''
-2
A
D''''
-3

Program 5 -- Discovering the Relationship between Two Circles in the Coordinate Plane

In this problem one is given two circles in a coordinate plane and must know the coordinates of the center and radius for each.

(X1,Y1) must be the coordinates of the center of the first circle.

R1 must be the radius of the first circle.

(X2,Y2) must be the coordinates of the center of the second circle.

R2 must be the radius of the second circle. This second circle must be the larger circle if one is larger.

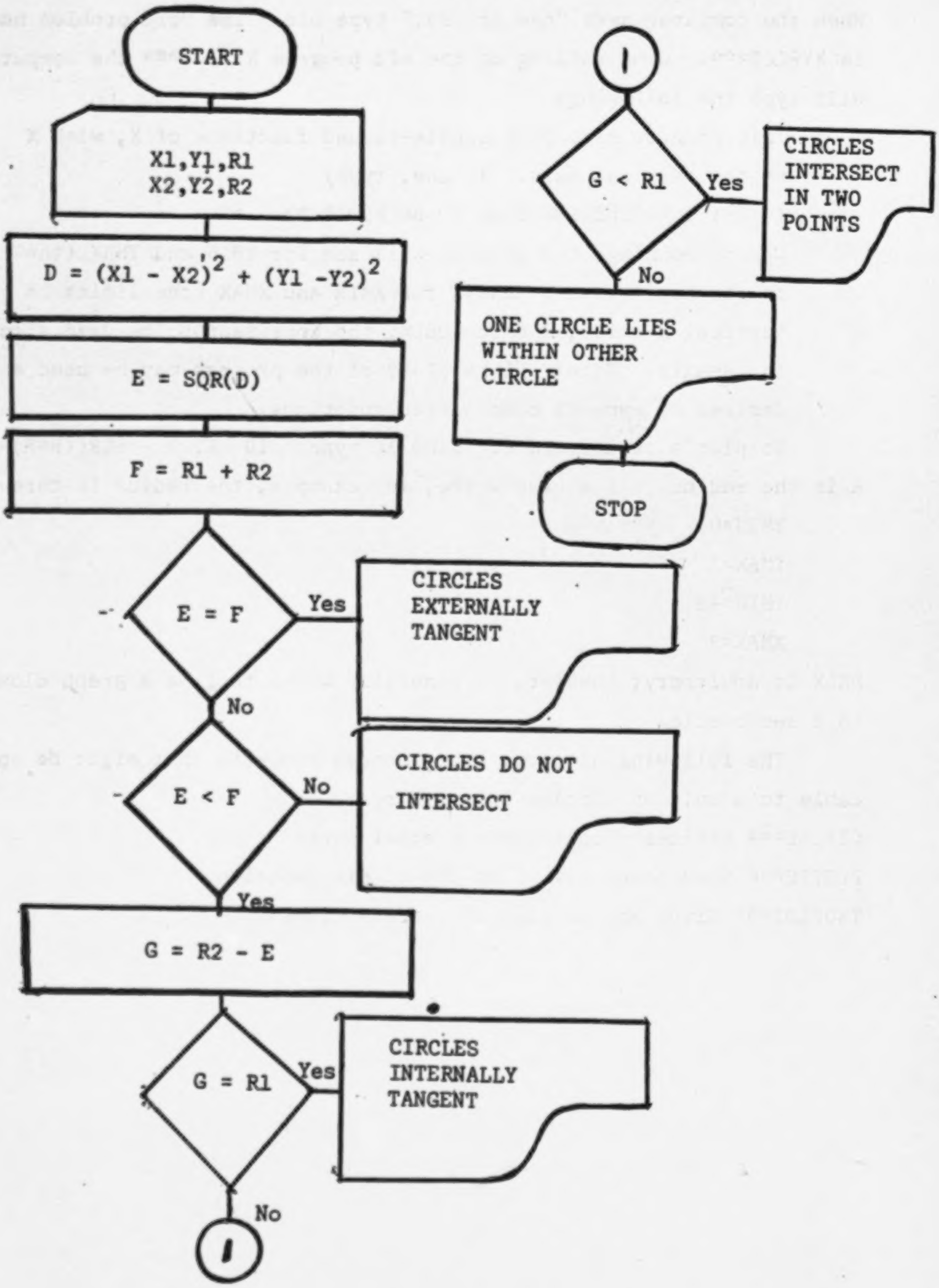
If the two circles have equal radii, then it does not matter which circle is first or second.

In the program E is the distance between the centers of the two circles. If the sum of the lengths of the radii is less than E, then the circles will not intersect. If, on the other hand, the sum of the lengths of the radii is greater than E, then four cases are possible as follows:

1) Circles are externally tangent if E is equal to the sum of the radii.
2) One circle lies within the other circle with no intersection if the larger radius minus E is greater than the other radius.
3) Circles are internally tangent if the larger radius minus E is equal to the other radius.
4) Circles intersect in two points if the larger radius minus E is less than the other radius.

```
 10 READ X1,Y1,R1,X2,Y2,R2
 20 LET D=(X1-X2)↑2+(Y1-Y2)↑2
 30 LET E=SQR(D)
 40 LET F = R1 + R2
 50 IF E = F THEN 90
 60 IF E<F THEN 110
 70 PRINT "CIRCLES DØ NØT INTERSECT"
 80 GØ TØ 10
 90 PRINT "CIRCLES ARE EXTERNALLY TANGENT"
100 GO TO 10
110 LET G = R2-E
120 IF G = R1 THEN 160
130 IF G<R1 THEN 180
140 PRINT "ONE CIRCLE LIES WITHIN OTHER CIRCLE WITH NO INTERSECTION"
150 GØ TØ 10
160 PRINT "CIRCLES ARE INTERNALLY TANGENT"
170 GØ TØ 10
180 PRINT "CIRCLES INTERSECT IN TWO POINTS"
190 GØ TØ 10
200 DATA
210 DATA
220 END
```

Flow Chart 5


START
X1,Y1,R1
X2,Y2,R2
D = (X1 - X2)^2 + (Y1 -Y2)^2
E = SQR(D)
F = R1 + R2
E = F
Yes
CIRCLES EXTERNALLY TANGENT
No
E < F
No
CIRCLES DO NOT INTERSECT
Yes
G = R2 - E
G = R1
Yes
CIRCLES INTERNALLY TANGENT
No
1
1
G < R1
Yes
CIRCLES INTERSECT IN TWO POINTS
No
ONE CIRCLE LIES WITHIN OTHER CIRCLE
STOP

Program 6 -- Plotting a Semicircle and Other Stored Programs

This problem is a stored program that will plot any semi-circle. When the computer asks "new or old," type old. The "old problem name" is XYPLOT***. Upon calling up the old program XYPLOT*** the computer will type the following:

```
This program sill plot single-valued functions of X, with X
on the vertical axis. To use, type:
10 LET Y = (THE FUNCTION TO BE PLOTTED)
During running, the program will ask for YMIN and YMAX (the limits
on the horizontal Y-axis), for XMIN and XMAX (the limits on the
vertical X-axis), and for DELX, the increment to be used along
the X-axis. Note: Lines 11-99 of the program may be used as
desired to express complicated functions.
```

To plot a semicircle for line 10 type: `10 LET Y - SQR((R*R)-(X*X))` R is the radius. In a case where, for example, the radius is three:

```
YMIN=0
YMAX=3
YMIN=-3
XMAX=3
```

DELX is arbitrary; however, .1 generally works to give a graph closest to a semicircle.

The following are some other stored programs that might be applicable to a unit on circles in geometry.

CIRCLE*** Divides circles into n equal parts
PLOTTO*** Simultaneously plots one to six functions
TWOPLOT*** Simultaneous plot of two functions

Program 7 -- Finding the Area of a Circle Incrementing with Narrow Vertical Rectangles

This program finds the area of a circular region incrementing with narrow vertical rectangles. In the program R is the radius of the circle, and S is the change in X or the width of each rectangle. For each vertical rectangle, then, the area will equal

length*width
= Y * X
= SQR(R^2-X^2)*(S)

For more accurate answers the program finds the value of Y halfway between any X_n and X_{n+1}. The program figures the area of each rectangle and adds it to the previous area (line 60). This area is 1/4 of the area of the circular region, and therefore in line 80 this area is multiplied by 4 to obtain the total area of the circular region. In other words, for a circle with center at the origin, it figures the area in the first quadrant and then multiplies that area by 4.

Two programs are listed here. The first will print the area after each rectangular area is added. The student can then see the area growing. For small increments the print part of this program gets to be quite lengthy. Therefore, the second program prints only the final area. In this program it is interesting to use S = .1, .01, .001 respectively to see the total area get closer and closer to the answer one obtains by merely multiplying Π*R*R, the usual formula for the area of a circular region.

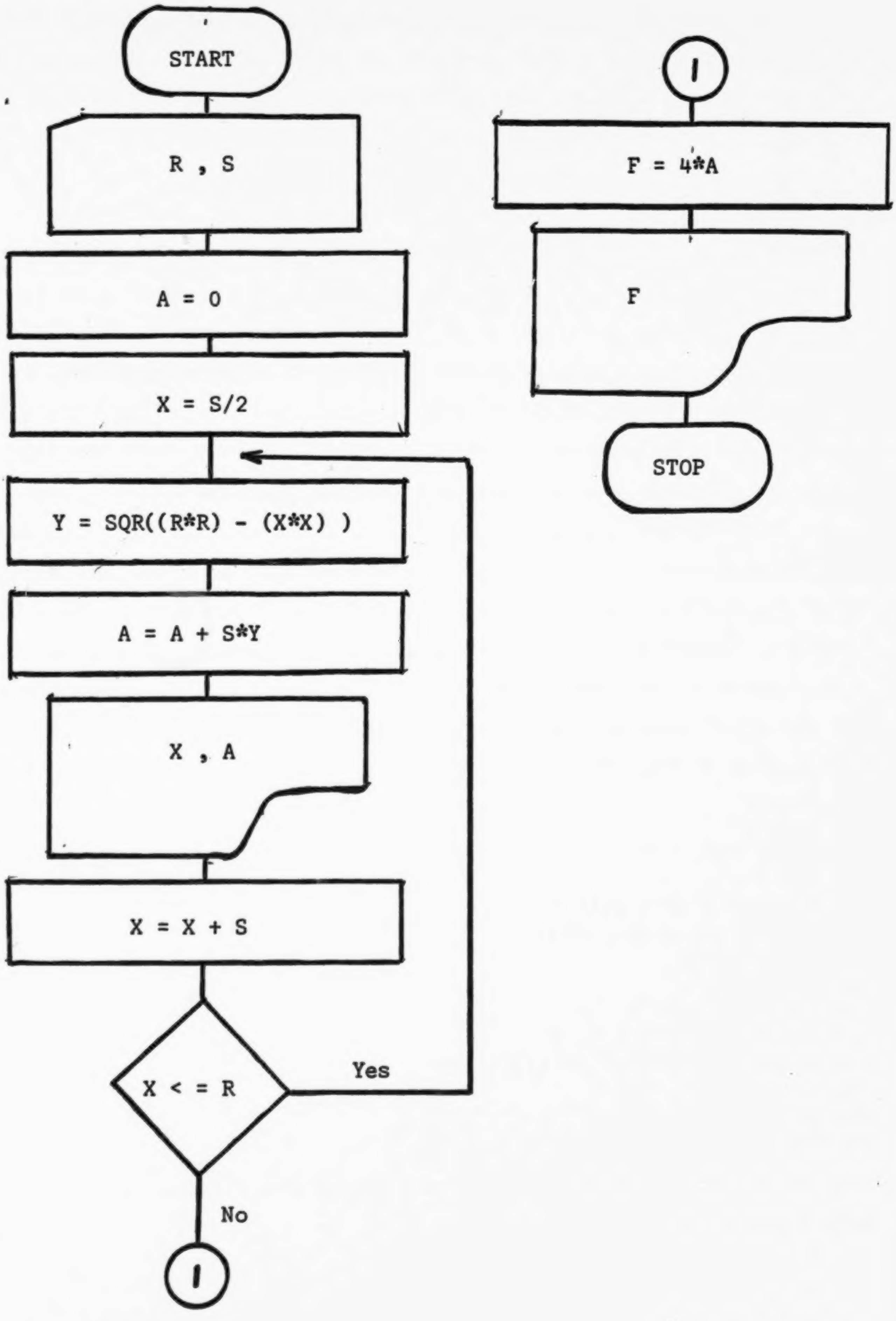
```
 10 READ R
 20 READ S
 25 PRINT "X", "A"
 30 LET A = 0
 40 FOR X=S/2 TØ R STEP S
 50 LET Y=SQR((R*R)-(X*X))
 60 LET A=A+S*Y
 65 PRINT X,A
 70 NEXT X
 80 LET F-4*A
100 PRINT "TOTAL AREA OF CIRCLE="F
110 DATA
120 DATA
150 END
```

Program for printing only the total area of the circle.

Omit lines 25 and 65.

FLOW CHART # 7

Flow Chart 7


START
R , S
A = 0
X = S/2
Y = SQR((R*R) - (X*X))
A = A + S*Y
X , A
X = X + S
X < = R
Yes
No
1
1
F = 4*A
F
STOP